SYSTEMATIC ANALYSIS OF METABOLISM AND TRANSCRIPTION WITH HIGH-THROUGHPUT SEQUENCING

By

YUANCHAO ZHANG

A dissertation submitted to the

School of Graduate Studies

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Doctor of Philosophy

Graduate Program in Microbiology and Molecular Genetics

Written under the direction of

Deanne M. Taylor

And approved by

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

New Brunswick, New Jersey

January, 2020
ABSTRACT OF THE DISSERTATION

Systematic analysis of metabolism and transcription with high-throughput sequencing

by YUANCHAO ZHANG

Dissertation Director:
Deanne M. Taylor

High-throughput sequencing (HTS) technology enables cost-effective and efficient measurement of the nucleobase sequences of millions of DNA molecules. In order to apply HTS in systematic investigation of metabolic variations and transcription initiation, I developed data analysis pipelines, mathematical methods, and software packages. Using these tools, we have identified sequence determinants in transcription initiation and pausing, which provide mechanistic insights into the interactions between RNA polymerase and DNA template. Also, we have modeled the metabolic variations across different tissues using single-cell RNA-seq data, which provides a novel computational method to infer metabolic variations from the differences in gene expression profiles.
ACKNOWLEDGEMENT

I deeply thank Dr. Deanne Taylor and Dr. Bryce Nickels’ guidance and support throughout my entire dissertation research. I also thank my thesis committee members Dr. Qian Cai and Dr. Yana Bromberg for their comments and suggestions to the progress of my dissertation research.

I am greatly thankful to all my collaborators in Dr. Bryce Nickels’ group, Dr. Douglas C. Wallace’s group, and Dr. Xilma Ortiz-Gonzalez’s group. Their generous help allows me to learn experimental techniques and explore exciting research ideas.

I truly appreciate the demanding administrative efforts made by Dr. Deanne Taylor, Mark Porter, Dr. Janet Alder, Diane Murano, Dr. Andrew Vershon, Tina Marottoli, Stephanie Otte, Hannah Lazarus, and many others. It was there attentive efforts that allow me to focus on my research.

This dissertation contains my previous publications and manuscripts prepared for future publications. The third chapter, “Systematic analysis of cellular metabolism using single-cell RNA-seq”, compiles the publications of work in collaboration with Dr. Bryce Nickels’ group (Vvedenskaya et al. 2015, 2016; Winkelman et al. 2016; Vvedenskaya et al. 2018), in which I developed computational tools and analyzed the high-throughput sequencing data with the guidance from Dr. Deanne Taylor and Dr. Bryce Nickels. The fourth chapter, “Scedar: a scalable Python package for single-cell RNA-seq exploratory data analysis”, is a pre-print deposited on bioRxiv pending publication (Yuanchao Zhang and Taylor 2019), in which I conceptualized the idea, developed the software package, performed benchmark analysis, and wrote the manuscript with the guidance from Dr.
Deanne Taylor and the suggestions from Dr. Yana Bromberg, Dr. Zhe Zhang, and Anthony Cros. The fifth chapter, “Systematic analysis of cellular metabolism using single-cell RNA-seq”, is a manuscript in preparation, in which I conceptualized the idea, developed the method, performed data analysis, and wrote the manuscript with the guidance from Dr. Deanne Taylor.
# TABLE OF CONTENTS

ABSTRACT OF THE DISSERTATION ........................................................................................................... ii

ACKNOWLEDGEMENT ................................................................................................................................. iii

TABLE OF CONTENTS ................................................................................................................................. v

LIST OF TABLES ........................................................................................................................................... x

LIST OF ILLUSTRATIONS ........................................................................................................................... xi

1 INTRODUCTION ........................................................................................................................................... 1

2 High-throughput sequencing technologies .............................................................................................. 6
   2.1 Table of Contents ................................................................................................................................... 6
   2.2 Introduction ......................................................................................................................................... 6
   2.3 HTS template preparation strategies .................................................................................................. 8
   2.4 Computational methods of HTS data analysis .................................................................................. 12
   2.5 Massively systematic transcript end readout .................................................................................... 20
   2.6 Merodiploid native-elongating transcript sequencing ....................................................................... 24
   2.7 CapZyme-seq ..................................................................................................................................... 27
   2.8 Single-cell RNA sequencing .............................................................................................................. 30

3 Systematic analysis of bacterial transcription initiation using high-throughput sequencing ......................... 34
   3.1 Table of Contents ............................................................................................................................... 34
   3.2 Introduction ....................................................................................................................................... 34
   3.3 Methods ............................................................................................................................................ 38
3.3.1 Construction of MASTER Plasmid Libraries
3.3.2 Generation of RNAs from the pMASTER-lacCONS-N7 Library In Vitro
3.3.3 Generation of RNAs from the pMASTER-lacCONS-N7 Library In Vivo
3.3.4 High-Throughput Sequencing of RNA 5’ Ends (5’ RNA-Seq)
3.3.5 5’ mNET-seq
3.3.6 CapZyme-Seq
3.3.7 MASTER-N10-XL: crosslinking of Bpa-containing RNAP in RPo
3.3.8 MASTER-N10-XL: primer extension mapping of RNAP-DNA crosslinks in RPo
3.3.9 MASTER-N10-XL: isolation of primer extension products
3.3.10 MASTER-N10-XL: ligation of adaptor to 3’ end of primer extension products
3.3.11 MASTER-N10-XL: size selection of adaptor-ligated DNA products
3.3.12 MASTER-N10-XL: amplification of DNAs
3.3.13 MASTER-N10-XL: size selection of libraries
3.3.14 MASTER and MASTER-XL: high-throughput sequencing
3.3.15 MASTER Data Analysis
3.3.16 5’ mNET-seq Data Analysis
3.3.17 CapZyme-Seq Data Analysis
3.3.18 MASTER and MASTER-XL: analysis of DNA templates
3.3.19 MASTER-N10 and MASTER-N7: analysis of TSS selection
3.3.20 MASTER-N10-XL: analysis of the positions of the RNAP trailing edge and the RNAP leading edge

3.4 Results
3.4.1 MASTER Analysis of TSS Selection Using Non-supercoiled Linear DNA
3.4.2 MASTER Analysis of TSS Selection Using Negatively Supercoiled DNA
3.4.3 MASTER Analysis of Effects of NTP Concentrations on TSS Selection
3.4.4 MASTER Analysis Defines the Extent of Productive Slippage Synthesis during Transcription Initiation
3.4.5 MASTER Analysis Reveals Effects of TSS-Region Sequences on Transcript Yield.
3.4.6 5’ mNET-seq analysis of Sequence-Specific RNAP–CRE Interactions in E. coli Transcriptome In Vivo ................................................................. 81

3.4.7 CapZyme-Seq Analysis of NCIN Capping with NAD’ In Vitro ................................. 84

3.4.8 Determinants for TSS Selection in NCIN Capping with NAD’ In Vitro .................. 87

3.4.9 Promoter Sequence Determinants for NCIN Capping with NAD’ In Vitro .......... 90

3.4.10 Promoter Consensus Sequence for NCIN Capping with NAD’ In Vitro .............. 93

3.4.11 Promoter-Sequence Determinants for NCIN Capping In Vivo .......................... 95

3.4.12 Identification and Quantitation of NCIN-Capped sRNAs In Vivo ....................... 98

3.4.13 MASTER analysis of discriminator sequence determinants on TSS selection..... 99

3.4.14 MASTER analysis of RNAP leading-edge and trailing-edge positions in transcription bubble ................................................................................... 103

3.5 Discussion .............................................................................................................. 109

3.5.1 MASTER analysis provides a comprehensive description of transcription initiation 109

3.5.2 MASTER as a new approach for comprehensive analysis of the relationship between nucleic-acid sequence and functional output ................................................. 110

3.5.3 Sequence-Specific RNAP–CRE Interactions in TSS Selection .......................... 111

3.5.4 Proposed Basis for Promoter Sequence Specificity for NCIN Capping with NAD’. 112

3.5.5 Scrunching in transcription start site selection .................................................. 114

4 Scedar: a scalable Python package for single-cell RNA-seq exploratory data analysis

116

4.1 Table of Contents .................................................................................................. 116

4.2 Introduction .......................................................................................................... 117

4.3 Methods ................................................................................................................. 120

4.3.1 scedar package design .................................................................................. 120

4.3.2 Minimum description length iteratively regulated agglomerative clustering .... 121

4.3.3 Minimum description length .......................................................................... 129

4.3.4 Mathematical theories on high-dimensional data analysis ............................ 133
5.4.1 Tissue and cell type specific metabolic models ......................................................... 190
5.4.2 Simulated tissue specific NAD$^+$ biosynthesis fluxes ............................................ 195
5.4.3 Metabolic variations within tissue and cell types estimated by bootstrapping ........ 198
5.4.4 Metabolic states within brain non-myeloid cell types .............................................. 202
5.4.5 Simulated metabolic fluxes are reproducible using different scRNA-seq datasets . 208

5.5 Discussion ...................................................................................................................... 213

6 Conclusion ........................................................................................................................ 217

BIBLIOGRAPHY .................................................................................................................. 219
LIST OF TABLES

Table 4.1. Real scRNA-seq datasets for benchmark..................................................146

Table 4.2. Number of clusters for benchmark..........................................................149

Table 5.1. Linear correlations between empirically measured tissue specific NAD+ biosynthesis fluxes and log2 mean read counts + 1 of key enzymes in NAD+ biosynthesis pathway. ..........................................................................................................198

Table 5.2. Simulated NAD+ biosynthesis fluxes in different datasets. .........................213
LIST OF ILLUSTRATIONS

Figure 2.1. Examples of HTS data visualization methods. ................................................................. 18
Figure 2.2. Massively Systematic Transcript End Readout: MASTER. ............................................. 22
Figure 2.3. 5’ mNET-seq analysis of *E. coli* transcriptome in vivo ............................................... 25
Figure 2.4. CapZyme-Seq, a High-Throughput-Sequencing Method to Detect NCIN-Capped RNA. ................................................................................................................................. 29
Figure 3.1. TSS Selection on a Non-supercoiled Linear DNA Template In Vitro. ......................... 66
Figure 3.2. TSS Selection on Negatively Supercoiled DNA Templates. ............................................. 70
Figure 3.3. Effects of NTP Concentrations on TSS Selection In Vitro ............................................. 73
Figure 3.4. Comprehensive Analysis of Productive Slippage Synthesis. ........................................ 77
Figure 3.5. Effects of NTP Concentrations on Transcript Yields In Vitro ....................................... 81
Figure 3.6. Effects of disrupting RNAP–GCRE interactions in vivo .............................................. 82
Figure 3.7. CapZyme-Seq Analysis of NCIN Capping with NAD+ In Vitro ..................................... 85
Figure 3.8. Determinants for TSS Selection in NCIN Capping with NAD+ In Vitro .................... 89
Figure 3.9. Promoter Sequence Determinants for NCIN Capping with NAD+ In Vitro ............. 92
Figure 3.10. Promoter Consensus Sequence for NCIN Capping with NAD+ In Vitro ........................ 94
Figure 3.11. CapZyme-Seq Analysis of NCIN Capping In Vivo ....................................................... 97
Figure 3.12. Sequences upstream of TSS region affect TSS selection ............................................ 102
Figure 3.13. Sequences upstream of TSS region affect RNAP trailing-edge/leading-edge distance ................................................................................................................................. 107
Figure 3.14. As TSS changes, RNAP leading-edge position changes, but RNAP trailing-edge position does not change. ................................................................. 108
Figure 4.1. Skewed division of hierarchical agglomerative clustering tree ..................................... 137
Figure 4.2. Demo of scedar. ............................................................................................................. 153
**Figure 4.3.** Scedar analysis of the scRNA-seq dataset containing 44,808 mouse retina cells generated by Drop-seq platform published by Macosko et al. (Macosko et al. 2015).

**Figure 4.4.** Scedar analysis of the scRNA-seq dataset containing 68,579 human peripheral blood mononuclear cells generated by 10x genomics GemCode platform published by Zheng et al. (Zheng et al. 2017).

**Figure 4.5.** Clustering method benchmarks on experimental datasets.

**Figure 4.6.** MIRAC and KNN rare transcriptomic profile detection results of experimental datasets.

**Figure 4.7.** Stability of clustering methods on experimental dataset.

**Figure 4.8.** Gene dropout imputation method benchmarks.

**Figure 4.9.** Benchmark results of gene dropout imputation methods.

**Figure 4.10.** KNN rare transcriptomic profile detection on the Zeisel et al. (Zeisel et al. 2015) dataset.

**Figure 4.11.** Identified genes separating the MIRAC clusters 1, 15, and 22 of the Zeisel et al. (Zeisel et al. 2015) dataset.

**Figure 5.1.** Example metabolic model.

**Figure 5.2.** Flux balance analysis.

**Figure 5.3.** Tissue and cell type context-specific metabolic models constructed using Tabula Muris scRNA-seq dataset (Schaum et al. 2018).

**Figure 5.4.** UMAP 2D embeddings of the transcriptomic profiles of different cells in Tabula Muris dataset.

**Figure 5.5.** General statistics of the Tabula Muris scRNA-seq dataset and metabolic models.

**Figure 5.6.** FBA simulated NAD⁺ biosynthesis fluxes.

**Figure 5.7.** NAD biosynthesis pathway in mouse.
Figure 5.8. FBA simulated NAD⁺ biosynthesis fluxes with bootstrapping.........................200

Figure 5.9. Bootstrapping results of metabolic modeling.................................................202

Figure 5.10. Metabolic states within mouse brain non-myeloid cell types......................205

Figure 5.11. UMAP 2D embeddings of the transcriptomic profiles of different brain non-
myeloid cell types in Tabula Muris dataset.................................................................206

Figure 5.12. Statistics distributions of the fluxes of brain non-myeloid cells..................208

Figure 5.13. Simulated cell-type specific metabolic fluxes using scRNA-seq datasets
from Tabula Muris and Allen Brain Mouse Atlas. .........................................................211
1 INTRODUCTION

High-throughput sequencing (HTS) has been applied extensively to biological research (Reuter, Spacek, and Snyder 2015). Its direct application is to determine the nucleotide base sequences of a mixture of DNA molecules in a massively parallel manner, e.g. the sequences of more than 100 million DNA molecules can be determined by Illumina NextSeq 550 sequencing in 15 hours. The abundance of each DNA species in the mixture can be inferred by counting the determined sequences. The capability of HTS in determining the abundances of DNA species is exploited in more than a hundred experimental methods to systematically quantify an enriched subset of nucleic acid species in cells, e.g. mRNAs quantified in RNA-seq (Nagalakshmi et al. 2008), mRNA regions bound by ribosomes quantified in Ribo-Seq (Ingolia et al. 2009), and chromosomal DNA regions with open chromatin in ATAC-seq (Buenrostro et al. 2013).

The applications of HTS in biological research also rely on various computational tools. The tools are designed to process the measured DNA sequences into biologically meaningful measurements, e.g. mRNA levels in RNA-seq, ribosome binding locations in Ribo-Seq, and chromosomal regions with open chromatin in ATAC-seq. This procedure is generally implemented by searching HTS measured sequences from a large collection of known nucleotide sequences, e.g. known DNA sequences in chromosomes, known RNA sequences, and designed plasmid sequences. The biologically meaningful measurements are further analyzed to infer biological insights, such as significant gene expression level differences between mutant and wildtype cells.
In order to apply HTS to study bacterial transcription regulation and metazoan metabolism, I have developed five computational tools to analyze the data generated by MASTER (Vvedenskaya et al. 2015), CapZyme-seq (Vvedenskaya et al. 2018), mNET-Seq (Vvedenskaya et al. 2014), and high-throughput single-cell RNA-seq (Macosko et al. 2015; Zheng et al. 2017). Using these tools, I have systematically characterized the transcriptional properties of promoter region sequences in *Escherichia coli*, and I have systematically simulated the metabolism of different mouse cell and tissue types. From these characterizations and simulations, I have obtained novel biological insights of the biochemical mechanism of *E. coli* RNA polymerase and metabolic variations across mouse cell types and tissues.

Transcription is an extensively regulated biological process in cells to synthesize RNA polymers based on DNA templates, which is initiated by the binding of RNA polymerase (RNAP) to DNA. The location of transcription initiation is partially regulated by the affinity between RNAP and DNA. Transcription is more likely to be initiated at DNA regions with higher affinity to RNAP. The affinity of a DNA region to RNAP is determined by its nucleotide base sequence, and some bases within the RNAP binding region have stronger impact on the affinity. In addition to the likelihood of initiating transcription, the base sequence of RNAP binding region also affects many other behaviors of transcription, such as the selection of transcription start site (Vvedenskaya et al. 2016, 2015) and the preference of initiating nucleotide species (Vvedenskaya et al. 2018).

Using HTS based experimental methods, we systematically profiled the transcription initiation properties of different promoter region sequences in *E. coli*. The profiled promoter region sequences are either on bacterial chromosomes or synthesized plasmids, which are transcribed in vitro or in vivo under different biological conditions,
such as RNA polymerase mutations, nucleotide triphosphate concentrations, and NAD⁺
decapping enzymatic treatments. The nascent RNA transcripts emanated from these
promoters under different conditions are collected to prepare high-throughput
sequencing libraries. From the sequencing reads of the nascent RNA transcripts, we
inferred various biochemical properties of different promoter region sequences, including
transcription start sites, relative transcript yields, the extent of productive slippage
synthesis, and the fractions of RNA transcripts with 5’ NAD⁺ capping (Vvedenskaya et al.

Cellular metabolism encompasses the chemical changes that occur in cells through a
myriad of chemical reactions, which is partially regulated by the expression of genes in
cells. The chemical reactions in cells are mediated by various gene expression products,
and their expression levels are regulated in cellular growth, proliferation, and adaptation
in different environments (Zhu and Thompson 2019). In a multicellular organism, cells
are differentiated into different types to exert distinct functions. In order to efficiently
support the cellular functions, the expression levels of metabolism related genes are
regulated by transcription factors, DNA methylations, and chromatin modifications (Lu
and Thompson 2012; Desvergne, Michalik, and Wahli 2006). Metabolism also regulates
the expression of genes by synthesizing substrates for DNA methylation and chromatin
modifications (Reid, Dai, and Locasale 2017). The interplay between metabolism and
gene expression maintains the normal status and function of the cells, and aberrations in
the interplay may disrupt the cellular homeostasis and cause diseases (DeBerardinis
and Thompson 2012).

In order to characterize the metabolic variations in different cell types, we used single-
cell RNA-seq (scRNA-seq) data to model the metabolism in each cell type. The
modeling procedure constructs a network of metabolic reactions based on the transcription levels of enzymes. From a curated genome scale metabolic model (GEM) that contains a comprehensive collection of metabolic reactions that can occur in an organism, the modeling procedure removes reactions that are catalyzed by enzymes with low or zero transcription levels in a cell type, which generates a network of reactions that are likely to exist in the modeled cell type. Using this cell type specific network, we simulate the rates of the metabolic reactions to infer the activity levels of metabolic pathways such as glycolysis, citric acid cycle, and oxidative phosphorylation. Applying this computational method to the Tabula Muris mouse scRNA-seq dataset (Schaum et al. 2018), we simulated the activity levels of nicotinamide adenine dinucleotide (NAD) biosynthesis in various mouse tissues, and the simulated levels are significantly correlated with empirically measured levels (Mori et al. 2014). Additionally, we also defined metabolic states in mouse brain non-myeloid cells using a dimensionality reduction method UMAP (McInnes and Healy 2018), and we interpreted the defined metabolic states using the metabolic model of each cell type.

Although HTS is capable of measuring various aspects of biomolecular behaviors (Reuter, Spacek, and Snyder 2015), a comprehensive understanding of an organism as a holistic system of molecules still requires revolutionary high-throughput technologies that are capable of measuring more aspects of biomolecular behaviors in a much more efficient manner. Although HTS can measure multiple samples at the same time, it is still not very cost-effective to measure multiple samples at multiple time points. Also, HTS measurements require nucleotide extraction from the sample, so it is difficult to measure the same sample at multiple time points or for multiple molecular behaviors. Furthermore, HTS is limited to nucleotide related molecular behaviors, but numerous molecules are not closely associated to nucleotides within an organism. In addition, the
localization and motion of molecules can only be measured by real-time microscopy in a low-throughput manner due to the lack of technologies to measure various molecular species without labeling or to label molecules in a high-throughput manner.
2 High-throughput sequencing technologies

2.1 Table of Contents

- Introduction
- HTS template preparation strategies
- Computational methods of HTS data analysis
- Massively systematic transcript end readout
- Merodiploid native-elongating transcript sequencing
- CapZyme-seq
- Single-cell RNA sequencing

2.2 Introduction

High-throughput sequencing (HTS) refers to the experimental procedures to efficiently determine the nucleotide base sequences of a large collection of DNA molecules. The experimental procedure generally consists of two consecutive phases, DNA template preparation phase and sequencing phase (Metzker 2010). In the template preparation phase, a collection of DNA molecules with special chemical properties, such as a fixed 5’ or 3’ end nucleotide base sequence, are prepared manually, and these prepared DNA molecules are called templates. Then, the nucleotide base sequences of DNA templates are determined in the sequencing phase, which is performed by an apparatus that is designed to automate the repetitive sequence determination procedure, and the apparatus is called sequencer. For example, illumina NextSeq sequencer is capable of sequencing up to 400 million DNA molecules within 30 hours.
Depending on the specific procedure of template preparation phase, the determined DNA nucleotide bases sequences contain different biologically meaningful information, such as genomic DNA sequences and gene expression levels. In whole genome sequencing, the DNA templates are prepared by amplifying the genomic DNA of an organism, so the determined DNA sequences are the fragments of genomic DNA molecules. In RNA-seq, the DNA templates are prepared by reverse transcribing the mRNA of cells, so the determined DNA sequences represent the quantity and diversity of mRNA species in the cells.

The sequencing results of DNA templates are analyzed by various computational tools to obtain biologically meaningful measurements. The most common analysis procedure is to compare the DNA template sequences with a database of known sequences. For example, the sequences of template obtained from reverse transcription of mRNA are compared with known transcript sequences or whole genome sequence to identify the original mRNA species of each template. Then, the number of templates is used to quantify the transcription level of the mRNA species.

In order to quantify the biochemical properties of bacterial transcription initiation, we developed three HTS applications, MASTER (massively systematic transcript end readout), CapZyme-seq and mNET-seq (merodiploid native-elongating transcript sequencing) (Vvedenskaya et al. 2015, 2014, 2018). These three HTS applications are able to capture the 5’ end sequence of RNA transcripts, which can be analyzed to quantify the transcription start site (TSS) preference of RNA polymerase (RNAP) bound to different promoter region sequences.
Apart from transcription initiation, we are also interested in the variability of cellular functions in metazoa, so we developed several computational methods to infer cellular metabolic reaction activities from single-cell RNA-seq datasets. Single-cell RNA-seq has been applied to profile multiple metazoan organisms during and after embryonic development (Schaum et al. 2018; Cao et al. 2017, 2019; Han et al. 2018; Aizarani et al. 2019). Based on the enzyme transcription levels in different cell types, we infer the fluxes of metabolic reactions through constraint based metabolic modeling (Schultz and Qutub 2016).

In this chapter, I will summarize the commonly used strategies for HTS template preparation and computational data analysis. Then, I will briefly describe the general experimental and computational procedures for MASTER, CapZyme-seq, mNET-seq, and scRNA-seq. The details of these three HTS applications are further described in the following chapters.

2.3 HTS template preparation strategies

The template preparation procedures are extremely diverse and flexible, which are capable of generating various types of biologically meaningful information through sequencing. In over 50 template preparation procedures (Reuter, Spacek, and Snyder 2015), the recurring strategies in the preparation of target nucleic acid molecules are enrichment, barcoding, and in situ modification. The target nucleic acid molecules are further prepared into DNA templates for a specific sequencer through adapter ligation and amplification.
Nucleic acid enrichment refers to the general procedure to isolate a specific type of nucleic acid molecules. For example, RNA molecules with poly-A tails are isolated by oligo-(dT) magnetic beads through annealing and magnetic purification. By isolating target nucleic acid molecules before the sequencing phase, the results will contain much fewer reads that are confounding or not informative, which would significantly increase the signal-to-noise ratio. Commonly used enrichment methods are immunoprecipitation of nucleic acid binding proteins, designing oligo sets that have higher affinity to target sequences, and nucleic acid size selection.

Nucleic acid barcoding is the experimental procedure to add a random DNA oligo to a specific location of each target DNA molecule, so that the random oligo base sequences in the sequencing results can be used to differentiate target DNA molecules. The target DNA molecules are mainly barcoded at two resolution levels:

1. Single DNA molecule level. In this resolution level, each target DNA molecule obtains a unique DNA oligo sequence, so that multiple sequencing results with the same oligo sequence are amplified from the same target DNA molecule. In practice, different target DNA molecules have a slight chance of obtaining the same oligo tag sequence (Fu et al. 2018), due to duplicated sequences in oligo pools and sequencing errors. In order to have fewer duplicated oligo sequences, the number of distinct random oligo sequences needs to be greatly larger than the number of target DNA molecules, which is usually achieved by synthesizing longer random oligo sequences. In order to reduce the sequencing error rate, the oligo tags are designed to locate in the region that has relatively stable quality and low error rates when sequenced by a specific sequencer. The barcoding at single DNA molecule level is usually performed by a DNA ligation step with a
synthesized pool of DNA oligos with random degenerately synthesized regions mixed with target DNA molecules.

2. Single experimental condition level. In this resolution level, each pool of target DNA molecules with the same experimental condition obtains a unique random oligo sequence, so that the oligo sequence in the sequencing results differentiates the experimental conditions. In a broad sense, experimental condition not only refers to genetic background and various treatments, but it also means any meaningful separation of target DNA molecules such as cellular origin (Macosko et al. 2015). The barcoding procedure at this resolution level can be performed either one-by-one on each pool of target DNA molecules or in a massive parallel manner. If performed one-by-one, DNA templates with the same experimental condition are ligated with a unique oligo sequence. Massive parallel barcoding relies on the micro-droplet scale separation of nucleic acid in different experimental conditions, exploiting either microfluidic controlling devices (Macosko et al. 2015; Zheng et al. 2017) or cellular membranes (Cao et al. 2017, 2019).

a. Specially designed microfluidic controlling devices are capable of partitioning a single cell and a single bead, covered with oligos having the same unique sequence, into a single aqueous microdroplet containing the required substrates and enzymes for adding barcodes to the target nucleic acid molecules, and thousands of such aqueous microdroplets are separated by an immiscible oil phase to form an 2-phase emulsion system (Brouzes et al. 2009).

b. Rather than using a microfluidic controlling device to partition barcodes and target molecules into a single droplet, cellular membranes can also be exploited to separate nucleic acid in a massive parallel manner. In the
single-cell combinatorial indexed sequencing (SCI-seq) developed by Vitak et al. (Vitak et al. 2017), isolated nuclei are placed in cell culture plates and reversibly permeabilized, and multiple nuclei in each well are barcoded with the same oligo sequence. Then, the nuclei are pooled together, and the nuclei separation and barcoding procedure is repeated for one or more times. When the number of distinct barcodes and barcoding repeats is relatively large, the chance of two different cells barcoded with the same sequence is very small, so that the barcode sequence is sufficient to identify the cellular origin of the target molecules.

Nucleic acid in situ modification refers to the physical, chemical, and enzymatic modifications of nucleic acid molecules on their original sites. These modifications can be detected in the sequencing results to obtain biological meaningful information. For example, bisulfite conversion is a chemical treatment that converts cytosine to uracil while leaving 5-methylcytosine intact, so the DNA methylation can be detected by comparing the sequencing results of target molecules with or without bisulfite conversion treatment (Frommer et al. 1992). CRISPR genome editing system can also be applied to change the DNA sequence of a specific genomic locus, so that the functional impact of the DNA sequence changes can be further detected (Adamson et al. 2016). Apart from changing DNA sequences, crosslinking DNA fragments that are spatially near each other can also be detected by checking whether a sequencing read contains the DNA sequences from multiple genomic loci (Schmitt, Hu, and Ren 2016).

The target molecules are further prepared for specific sequencers through adapter ligation and amplification. Adapters are oligos with known base sequences that are complementary to the oligos immobilized on the surface of the sequencing reaction
chamber. Adapters are ligated to the target molecules in order to fix them on the sequencing reaction chamber through replication, with the immobilized oligos as primers for the replication. The product DNA molecule is composed of a target molecule strand and a synthesized strand, and the synthesized strand has one end immobilized on the surface of the sequencing reaction chamber and the other end free to anneal to other immobilized oligos, because the adapters are ligated to both ends of a target molecule. Therefore, the replication procedure can be repeated multiple times, which would produce a local cluster of immobilized replicates of the target molecule, in order to enhance the signal of sequencing reactions. Before adding into the sequencing reaction chamber, target molecules are also amplified through polymerase chain reaction (PCR) or in vitro transcription, in order to optimize the number of target molecules for the sequencing run. If the number of target molecules is much higher than the sequencing chamber capacity, the sequencing reaction of each target molecule may interfere with each other to decrease the quality of the sequencing reads. If the number of target molecules is much lower than the sequencing chamber capacity, multiple barcoded samples can be sequenced at once to reduce the cost of the experiment.

2.4 Computational methods of HTS data analysis

HTS computational data analysis converts the determined sequences of DNA templates into biological meaningful measurements, and the measurements are further interpreted with domain specific knowledge to obtain biological insights. The recurring strategies in HTS computational data analysis include string level processing, alignment, counting, visualization, inference, and interpretation, which are usually performed sequentially.
HTS data analysis consists of a series of conversions from one type of data to another, and these conversions need to be implemented by analysts using programming languages, third-party binary tools, and pipeline management systems (Leipzig 2017). HTS data analysis usually cannot be performed only in graphical spreadsheet applications such as Microsoft Excel and Google Sheets for multiple reasons, including:

- Specialized tools are required to convert determined template sequences into biologically meaningful measurements.
- The size of the tables is usually too large to be handled efficiently.
- The inference procedures are too complex to be implemented with simple algebraic formula.

Therefore, HTS data analysis requires extensive programming, which utilizes third-party tools and databases to convert data types. The starting data type of HTS data analysis is the determined template sequences generated by sequencers. Through multiple steps of conversion, the template sequences will be converted into biological meaningful measurements such as gene transcription levels, splicing isoform fractions, and SNP ratios. These biologically meaningful measurements are further tabularized for statistical inferences, in order to compare different experimental conditions. The significant differences between different experimental conditions are further interpreted with domain specific knowledge through extensive literature reviews.

The determined template sequences are commonly processed as strings before considering their biological meanings. The procedure usually includes extracting sequence regions of interest, handling barcodes, and filtering low quality reads. Depending on the template preparation procedure, each sequencing read, i.e. the determined sequence of a DNA template, may contain multiple regions with different meanings, e.g. barcode region, biologically meaningful region, and adapter region. The
adapter regions are usually removed from the sequencing read before further analysis. The barcode region sequences are analyzed to remove amplification bias and determine experimental conditions. When barcodes are used to identify each individual target nucleic acid molecule, the sequencing reads with the same barcode sequence represent the same target nucleic acid molecule before amplification, so that the biologically meaningful region sequence will only have one record in the output file. When different barcodes are used to distinguish different experimental conditions, the sequence reads with the same barcode will be pooled into its individual output file to represent the results of the experimental condition. In addition to sequence processing, the quality scores provided by the sequencer are also commonly used to remove low quality sequencing reads. After processing the accessory regions, the biologically meaningful regions are output into one or more files for further analysis.

The biologically meaningful region sequences are compared to a reference collection of known sequences to determine the origin of each template, and the comparison procedure is usually called mapping or alignment. For example, in RNA-seq, the biologically meaningful region of a sequencing read is from mRNA molecules, and the original mRNA species of each sequencing read is determined by comparing the read sequence to the reference sequences of whole genome or transcriptome. The whole genome, exome, and transcriptome sequences are most commonly used reference sequence collections for determining the target molecule origin of sequencing reads, but there are also specialized reference sequence collections such as a set of synthesized plasmids with different sequences (Vvedenskaya et al. 2015). The comparison procedure essentially searches for the reference sequences that are similar to the sequencing read. Because the sequencing reads are not exactly the same as the reference sequence, the searching allows non-exact matches, which is performed with
various string alignment algorithms (Smith and Waterman 1981; Langmead et al. 2009; Langmead and Salzberg 2012; H. Li 2013; Dobin et al. 2012; D. Kim, Langmead, and Salzberg 2015). The computer program implementations of the searching procedures become standalone tools known as sequence aligner or mapper for HTS data analysis, such as bwa and bowtie (H. Li 2013; Langmead et al. 2009), because the searching procedure does not require customization based on specific template preparation procedures, and the program is usually too complex to quickly implement from scratch.

After aligning the sequencing reads to the reference collection of sequences, the number of reads aligned to specific regions of the reference sequences is counted to form a table, conventionally with rows representing different regions and columns representing different samples, for further analysis. The counting procedure depends on the biological meaning of the sequencing reads. For example, in whole genome sequencing, each sequencing read represents a fraction of the genomic DNA sequence, so the number of bases of aligned reads at each position of the reference sequences is counted to identify single nucleotide variants. In RNA-seq, each sequencing read represents a fraction of the mRNA sequence, so the number of reads aligned to each annotated gene is counted to quantify gene transcription levels. Multiple counting procedures can be performed on the same set of aligned reads. In whole genome sequencing, structural variants can be inferred by counting the number of reads that each aligned to distinct locations of reference sequences (Layer et al. 2014). In RNA-seq, the number of reads aligned to the exon-intron junctions can be counted to infer alternative splicing variants (Vaquero-Garcia et al. 2016). In general, the read counts represent the abundance of biologically meaningful features, which can be tabularized with rows as the biological features, columns as different samples, and entries as read counts for further analysis.
Visualization provides easy-to-perceiv biological intuitions and insights about the tabularized read counts, which are very difficult to obtain by directly looking over the whole table that have thousands to millions of entries. Therefore, visualization methods are applied to illustrate the overall and specific patterns in the table (Figure 2.1). The commonly used methods to illustrate the overall patterns include heatmap and dimensionality reduction:

- The large read count tables can be plotted as heatmaps, with clustered rows and columns, to show read count patterns clearly across different samples and biological features.

- The high-dimensional features or samples in the table are commonly plotted in lower-dimensional space through dimensionality reduction methods to visualize the similarity between different features or samples. Each sample in the table can be considered as a high-dimensional data point with each biological feature as an axis, and each feature in the table can also be similarly considered as a high-dimensional point with each sample as an axis. Because the number of axes is usually greater than 3, it is difficult to plot the raw data points with each feature or sample as an axis. In order to plot different features or samples, we can represent each sample or feature with a 2-dimensional point, and make sure that the distances between different points capture their relative similarity in the high-dimensional space. This mathematical procedure to represent high-dimensional data points with lower-dimensional data points is called dimensionality reduction, which essentially embeds high-dimensional data points in a lower-dimensional space, so the coordinates of the lower-dimensional points are also called the embeddings of high-dimensional data points.

The commonly used methods to illustrate the specific patterns include distribution and reference sequence plots.
- In distribution plots, we plot the distribution of read counts of a feature in different set of samples, in order to illustrate the variations between different set samples. The commonly used plot types are boxplot, violin plot, density plot, swarm plot, and histogram.

- In reference sequence plots, the read counts are plotted with the reference sequence, in order to illustrate the alignment differences between different samples. For example, the read counts can be plotted on the sequence of a gene, with abstract illustration of the exons and introns, so that the read count differences can be used to illustrate splicing events.

Although there are various ways to illustrate HTS data, the purpose is to provide clear biological intuitions and insights to the viewer.
The observed patterns in HTS dataset are further examined by statistical inference methods. The most commonly used inference method is to test whether the read counts of a specific feature are significantly different between two or more conditions, e.g. whether the read counts aligned to a gene is higher in the samples with genetic
mutations than those samples without mutations. When the number of performed tests is large, the test results need to be adjusted to reduce the false positive rate by controlling family-wise error rate or false discovery rate (Benjamini and Hochberg 1995; Farcomeni 2008). The intuition is that when the number of tests is very large, there will be some positive test results caused by the inaccuracy of the measurements rather than the underlying biological differences. Assuming that the inaccuracy results in a stochastic distribution of measurements centered on the true value, there is a slight chance of obtaining a positive test result, where two conditions have significantly different measurements, even though the true values are identical. Apart from hypothesis testing, statistical inference methods with latent variable models, especially hidden Markov models, are also commonly used in HTS data analysis. The basic assumption of such methods is that the measurements are generated by a statistical model with a set of biologically and technically meaningful latent variables. With this assumption, we are able to infer the most likely values of the latent variables based on the observations. Machine learning methods can also be applied to HTS data analysis for inference (Bzdok, Altman, and Krzywinski 2018). Machine learning methods rely on fewer and even no statistical assumptions of the data, and they can be applied to extremely large datasets, but the results are much harder to interpret due to the high complexity of methods.

Complex inference methods are also developed to identify underlying patterns in the data. These methods essentially evaluate multiple features at once to identify complex patterns, and they can only be applied on a specific type of HTS data. For example, gene set enrichment analysis (GSEA) identifies whether the transcription levels of a biologically meaningful set of genes is significantly different between samples in different experimental conditions (Subramanian et al. 2005). The transcription level of each
individual gene may not be significantly different, due to multiple testing corrections, small effect sizes, and measurement inaccuracy. In such cases, we would still be able to identify significant transcription level differences by finding consistent changes in multiple relevant genes. Many inference methods also take the relationships between different features into account, and the relationships are usually represented as networks with nodes as features and edges as their relationships. The most commonly used feature relationship is the correlation between different features in different experimental conditions, which provides an additional layer of information on top of the individual changes.

The statistically significant observations need to be interpreted in biological context formed by previous research findings. The interpretation is usually performed by reviewing previous literature, checking whether the observations are consistent with previous findings, identifying biologically meaningful questions that are not answered previously, and approaching the identified questions with current observations. Ideally, the interpretation is planned before doing the experiments, but it is usually changed and performed throughout research. Although the main purpose is to report novel findings in a specific biological context, it is also important to identify observations that are consistent with previous findings, because the consistent observations are able to validate the quality of the data and analysis procedure.

2.5 Massively systematic transcript end readout

We developed an HTS application, massively systematic transcript end readout (MASTER), to systematically investigate bacterial transcription initiation with different promoter region sequences (Figure 2.2). In template preparation procedure, we
construct a library of plasmids with randomized promoter region sequences, and these plasmids are transcribed in vitro or in vivo. The RNA transcripts are extracted to prepare templates for high-throughput sequencing. The sequencing results are analyzed to count the number of transcripts that started transcription at each position of the promoter region.
Figure 2.2. Massively Systematic Transcript End Readout: MASTER. Top: generation of pMASTER-lacCONS-N7 library. An oligodeoxynucleotide carrying the lacCONS-N7 promoter and 15-nt barcode sequence (blue) is used as template in a PCR reaction using primers that introduce BglII sites. The BglII digested PCR product is cloned into BglII digested plasmid pSG289 to generate plasmid pMASTER-lacCONS-N7, which contains $4^7$ (~16,000) sequences at positions 4–10 bp downstream of the lacCONS -10 element (green). Middle: product generated by emulsion PCR is used for high-throughput sequencing analysis to assign barcodes to TSS-sequence variants. PCR primers shown in red (5’ and 3’ adaptors) carry sequences that facilitate analysis using an Illumina HiSeq. Bottom: 5’ RNA-seq analysis of RNA produced from the library in vitro and in vivo. The sequence of the barcode is used to assign the RNA to a TSS region, the sequence of the 5’ end is used to define the TSS, and the number of reads is used to measure transcript yield from each TSS-region sequence.

In the constructed plasmid libraries, each plasmid contains a randomized 7-base promoter region (N7 in Figure 2.2) and a randomized 15-base barcode region downstream (N15 in Figure 2.2). The randomized promoter region provides various nucleic acid context for transcription initiation, and the barcode region identifies the promoter region sequence of each RNA transcript. The -10 and -35 elements (purple sequence regions in Figure 2.2) upstream of the randomized promoter region are conserved DNA sequences for RNA polymerase (RNAP) binding. The locations of -10 and -35 elements on the plasmid make transcription preferentially start at the 4th base of the randomized promoter region sequence. Therefore, most of the RNA transcripts do not contain the full 7bp promoter region sequence, but they contain the full 15bp barcode region sequence which is located about 20bp downstream of the promoter region. In
order to identify promoter region sequences using barcode region sequences, we also sequenced the promoter and barcode region of the plasmids.

We prepare two types of sequencing templates either from PCR amplified plasmid DNA or transcribed RNA, in order to determine the sequences of the constructed plasmids or their transcripts. One type of sequencing templates is prepared by PCR amplifying the DNA plasmid region that covers the randomized promoter and barcode regions, and their sequences are used to associate each promoter region sequence with a barcode sequence. The other type of sequence templates is prepared by reverse transcribing the RNA transcripts of the plasmids and PCR amplifying the reverse transcription products, and their sequences are used for determining the transcription start site of the RNA transcripts.

The computational data analysis procedure counts the number of transcripts that initiated transcription at each position of the randomized promoter region. The first step is to analyze the PCR amplified plasmid DNA sequences to unambiguously associate each promoter region sequence with one or more barcode region sequences. Then, for each RNA transcript sequence, its full promoter region sequence is determined by searching its barcode region sequence in the DNA sequence analysis results, and the transcription initiation position is determined by counting the number of bases between the first base of the sequencing read and the first base of the non-randomized region. The counts of transcription initiation positions are tabularized with each row as a promoter region sequence and each column as an initiation position.
2.6 Merodiploid native-elongating transcript sequencing

Merodiploid native-elongating transcript sequencing (mNET-seq) is an HTS application to characterize the biochemical properties of mutant RNAP in transcription (Vvedenskaya et al. 2014, 2016). The merodiploid E. coli contain a chromosome-encoded wildtype RNAP gene and a plasmid-encoded mutant or wildtype RNAP gene, and the plasmid-encoded RNAP is tagged with an epitope that can be used to enrich derived RNA transcripts by immunoaffinity purification (Figure 2.3). The enriched RNA transcripts are reverse transcribed and amplified to prepare high-throughput sequencing templates. The sequencing results are aligned to the E. coli reference genome, and the alignments are counted to characterize the transcription behaviors of mutant RNAP.
Figure 2.3. 5’ mNET-seq analysis of E. coli transcriptome in vivo. Steps in 5’ mNET-seq analysis of chromosomal promoters: (Top) RNAP derivatives in cells with either wild type subunit encoded by chromosomal rpo+ or 3xFLAG tagged mutant subunit encoded by plasmid rpo∗,3xFLAG; (Middle) RNAPs on the same transcription unit in four cells (RNA products in blue are associated with mutant RNAP); and (Bottom) isolation of RNA products after immunoprecipitation with anti-FLAG affinity gel and sequencing analysis
of RNA 5’ ends. In this example, TSS selection at genome coordinate labeled “a” is decreased with the mutant RNAP derivative.

The RNA transcripts derived from mutant RNAP are enriched by immunoaffinity purification of the epitope tagged RNAP. As the merodiploid *E. coli* also have a chromosome-encoded RNAP for creating an isogenic background, the RNA transcripts derived from plasmid-encoded RNAP needs to be enriched before further template preparation, in order to specifically characterize the biochemical behavior of plasmid-encoded RNAP. The enrichment is implemented by tagging the plasmid-encoded RNAP β subunit with 3xFLAG sequence, so that the plasmid-encoded RNAP can be purified by anti-FLAG gel through immunoaffinity. The RNA transcripts that are derived from the enriched RNAPs are reverse transcribed and amplified to prepare sequencing templates for specific illumina sequencers.

The sequencing reads are aligned to *E. coli* reference genome, and the alignments are counted to identify either transcription initiation site or RNAP pausing site. We used either bwa or bowtie for aligning the sequencing reads to the *E. coli* reference genome sequence (H. Li 2013; Langmead et al. 2009), depending on the specific alignment requirements. In order to count the 5’ or 3’ end positions of alignments, we developed a suite of command line tools, freely distributed at https://github.com/logstar/NET-seq-CLT, because the available tools cannot meticulously count alignment end positions or handle strandness. The biological meaning of the alignments depend on the template preparation procedure. If the 5’ end of RNA transcripts are selected for sequencing template preparation, the 5’ end of the alignments is the transcription start site (TSS) of each transcript. If the 3’ end of RNA transcripts are selected for sequencing template
preparation, the 3’ end of the alignments is the RNAP synthesizing edge of the
transcript. The local maxima of 5’ counts is a preferred TSS of the genomic locus, and
the local maxima of 3’ counts is a preferred transcription pausing site. The nearest
upstream and downstream genes of the transcription start and pausing sites can also be
identified by our command line tools, with the up- and down-stream directions of sites on
different strands correctly handled.

The read counts and sequences near transcription start and pausing sites are used to
infer the biochemical properties of wildtype and mutant RNAP. The DNA sequence
preferences of wildtype and mutant RNAP in initiating and pausing transcription can be
inferred from the genomic sequences of transcription start and pausing sites
respectively. The transcription start site sequences can also be used to infer the
transcription factors that are assembled for transcription initiation, according to the
known DNA sequence preferences of transcription factors (Haugen, Ross, and Gourse
2008). For each type of RNAP, we can generate two sequencing libraries for its 5’ and 3’
ends separately, so that the transcription start sites can be associated with the
downstream transcription pausing sites to characterize the effects of transcription factors
on RNAP pausing. The biochemical differences between wildtype and mutant RNAP
provide insights into the mechanisms of interactions between nucleotide sequence
elements and RNAP active sites in transcription initiation and pausing.

2.7 CapZyme-seq

CapZyme-seq is an HTS application to detect and quantify RNA transcripts capped by
nucleoside-containing metabolites at the 5’ end (Vvedenskaya et al. 2018). Nucleoside-
containing metabolites, such as nicotinamide adenine dinucleotide (NAD+), can be
incorporated as 5’ caps on RNA by serving as non-canonical initiating nucleotides (NCINs) for transcription initiation by RNA polymerase (RNAP), which have been identified in bacteria and eukaryotes (Bird et al. 2016; Cahová et al. 2015; Y. G. Chen et al. 2009; Kowtoniuk et al. 2009; Jiao et al. 2017). In order to detect NCIN-capped RNA, we treat RNA transcripts with NCIN-decapping enzymes NudC or Rai1 at the beginning of template preparation, so that NCIN-capped RNA 5’ end sequencing results will become one or two bases, respectively, shorter than the RNA molecules that are not 5’ capped by NCINs (Figure 2.4). We applied NCIN-decapping enzymatic treatments prior to MASTER and mNET-seq, and we compared the counts of sequencing read 5’ end position with or without the enzymatic treatment to infer the proportion of RNA transcripts with 5’ capped by NCINs for each promoter region sequence. We further defined the promoter region sequence patterns that favor 5’ NCIN capping in transcription initiation (Vvedenskaya et al. 2018).
Figure 2.4. CapZyme-Seq, a High-Throughput-Sequencing Method to Detect NCIN-Capped RNA. (A) Structures of ATP and adenosine-containing NCINs NAD⁺, NADH, dpCoA, and FAD. Red, identical atoms. (B) Processing of RNA 5’ ends by NudC, Rai1, and Rpp. Red, common moiety of each 5’ end; black, distinct moiety of each 5’ end; gray, remainder of RNA. (C) Products of processing of NCIN-capped RNA 5’ ends by NudC, Rai1, and Rpp. (D) CapZyme-seq procedure. Gray, RNA; purple, 5’ adaptor; red, 3’ adaptor; black cDNA.
2.8 Single-cell RNA sequencing

We developed two computational tools to analyze single-cell RNA-seq (scRNA-seq) data. The python package scedar provides an efficient and robust interface for scRNA-seq exploratory data analysis, such as visualization, clustering, and rare cell type identification. We also developed a method to model cell-type specific metabolism using scRNA-seq data through constraint based metabolic modeling.

The experimental and data analysis procedures are adapted from bulk RNA-seq. The purpose of scRNA-seq template preparation is to uniquely barcode the templates prepared from each individual cell. The barcoded templates are pooled together to sequence multiple cells in one run, and the sequencing reads are separated into individual cells according to their barcodes. The sequencing reads of each cell are aligned to the reference genome sequence of the organism, and the reads aligned to each gene are counted to generate a table of read counts with rows as genes and columns as cells. The read count table can be further analyzed infer various biological properties, mainly cellular transcriptomic heterogeneity, developmental trajectory, and novel cell types (Tanay and Regev 2017). The inferred biological properties are interpreted in relevant context to refine previous physiological and pathological understandings at single-cell level.

The overall goal of scRNA-seq template preparation procedure is to uniquely barcode the templates originated from the mRNA of each single cell. The mRNA transcripts are selected by poly(A) enrichment and further amplified by PCR to prepare sequencing templates. In order to uniquely barcode the sequencing templates of each individual cell, mainly two strategies are applied:
1. Physically isolate the template preparation and barcoding reactions of each individual cell (Macosko et al. 2015; Zheng et al. 2017; Picelli et al. 2014).

2. Stochastically sample multiple cells multiple times and barcode each sample with distinct barcodes at each time (Vitak et al. 2017).

It is natural and intuitive to physically isolate distinct cells to achieve unique barcoding, which has been mainly implemented by manual cell picking by microdissection (Picelli et al. 2014) and microfluidic partitioning of single cells into their own aqueous microdroplets (Macosko et al. 2015; Zheng et al. 2017). The second strategy exploits stochastic sampling to ensure that nearly all barcodes are unique to each individual cell. In the sampling procedure, the pool of all cells or nuclei is sampled multiple times without replacement, and each sample of cells is placed in one well of the cell culture plate and reversibly permeabilizing to perform template preparation and barcoding reactions. The barcodes are the same within each well and distinct between different wells. As the permeabilization is reversible, the cells can be pooled together after the barcoding reactions, and the sampling and barcoding procedure can be repeated to stochastically introduce more barcodes to each cell. When the number of sampling repetitions and the number of samples in each repetition are high enough, nearly all cells have a unique sequence of barcodes. For example, if the stochastic procedure samples 500 cells from 1000,000 cells for 3 times, the number of cells containing the same sequence of barcodes is 125.66 ± 16 (mean ± standard deviation), based on a computer simulation of the stochastic sampling procedure for 100 times.

The barcoded sequencing reads are aligned to the reference genome sequence of the organism, and the reads aligned to each gene are counted for each individual cell according to their barcodes, in order to generate a read count table with each row as a gene and each column as a cell. The alignment results can also be analyzed to
characterize transcription properties other than the number of transcripts in single cells, such as alternative splicing (Arzalluz-Luque and Conesa 2018).

Scatter plots of the dimensionality reduced scRNA-seq data are commonly used to obtain intuitions and insights about the transcriptomic heterogeneity of the sequenced cells (Figure 2.1). Dimensionality reduction essentially converts a column of the read count table, which contains the read counts of all genes for the corresponding cell, into another column with much fewer entries, and the entry values can represent the similarity of gene read counts between cells. After plotting the dimensionality reduced values of cells into scatter plots, we can look over the plots to see the separation of cells into different clusters.

Clustering analysis is applied on the dimensionality reduced scRNA-seq data to separate cells into different cell types or states. The cells are separated into distinct groups according to their dimensionality reduced values through clustering methods, and the groups of cells are called cell clusters and further analyzed to identify their cell types or states. Cell types are defined by previously identified marker genes. Within a cell type, different patterns of gene transcription levels are used to define cell states. Novel cell types and states may be identified through clustering analysis. The discrete clustering analysis can also be extended into a continuous analysis to infer biologically meaningful trajectories, such as development, differentiation, and proliferation (Saelens et al. 2019).

We developed a python package scedar (Single Cell Exploratory Data Analysis for RNA-seq), in order to facilitate the exploratory data analysis (EDA) of large-scale scRNA-seq datasets (Yuanchao Zhang and Taylor 2019). The package provides a convenient and reliable interface to perform visualization and clustering analysis. Additionally, scedar
also provides a few other methods to impute gene dropouts, detect rare cells, and identify cluster separating genes.

Apart from exploratory data analysis, we also developed a computational method to simulate the metabolic fluxes of different cell types under different conditions using single-cell RNA-seq (scRNA-seq) data and context specific constraint based modeling. The genome scale constraint based metabolic model of an organism is a curated set of metabolic reactions that could occur in the organism (King et al. 2016). From the genome scale metabolic model, we extract a subset of metabolic reactions to represent the metabolism of a cell type, based on the levels of enzyme expressions inferred from the scRNA-seq data of the cell type. The cell-type specific metabolic model could be used to simulate the optimal fluxes of metabolic reactions under steady state.
3 Systematic analysis of bacterial transcription initiation using high-throughput sequencing

Abstract: Transcription is a highly regulated biological process that synthesizes RNA based DNA templates. We systematically characterized the transcription initiation on genes and plasmids with different promoter sequences using high-throughput sequencing in *Escherichia coli*. We identified promoter sequence patterns that can determine the biochemical properties of RNA polymerase (RNAP) in transcription initiation, including transcription start site (TSS) site selection, transcript yields, transcriptional slippage, and utilization of non-canonical nucleotide for initiation. Apart from systematic characterizations, we further elucidated the mechanism of transcription initiation by showing that RNAP scrunches DNA prior to RNA synthesis for TSS selection.

3.1 Table of Contents

- Introduction
- Methods
- Results
- Discussion

3.2 Introduction

Defining the mechanism of any biological process involving protein-nucleic acid interactions requires a full understanding of the relationship between nucleic acid
sequence and functional output. Often such studies require the construction and analysis of many individual variants of a given DNA sequence. However, for processes involving extensive protein-nucleic acid interactions, a comprehensive analysis of the functional output derived from all sequence variants is not feasible with conventional approaches that test individual sequence variants on a one-by-one basis. Instead, such analyses require approaches that facilitate the parallel interrogation of thousands of individual sequence variants en masse (Melnikov et al. 2012; Patwardhan et al. 2012; T. Heyduk and Heyduk 2015; E. Heyduk and Heyduk 2014).

The process of transcription involves extensive interactions between RNA polymerase (RNAP) and a double-stranded DNA template. Each stage of transcription, and each reaction carried out by RNAP during transcription, is affected by the sequence context of the DNA template (Larson, Landick, and Block 2011; F. Wang and Greene 2011; Dangkulwanich et al. 2014; Ruff, Record, and Artsimovitch 2015; Washburn and Gottesman 2015). While structural studies have revealed some of the RNAP-nucleic acid interactions that modulate transcription (Murakami 2015), a quantitative understanding of how DNA sequence influences transcription output requires comprehensive knowledge of RNAP activity in all sequence contexts.

We developed three high-throughput sequencing (HTS) applications to facilitate the comprehensive analysis of the relationship between nucleic acid sequence and functional output during transcription for *Escherichia coli* RNAP, including:

- Massively systematic transcript end readout (MASTER) enables parallel functional analysis of $4^7$ (~16,000) and $4^{10}$ (~1,000,000) distinct plasmid DNA template sequence variants in vitro and in vivo (Vvedenskaya et al. 2015).
- CapZyme-seq pararelly detects 5' nucleoside-containing metabolite capping in RNA molecules that are transcribed from 4^7 distinct plasmid DNA template sequence variants in vitro and in vivo (Vvedenskaya et al. 2018).

- Merodiploid native-elongating-transcript sequencing (mNET-seq) enables parallel characterization of the 5' and 3' ends of the RNA products generated from all E. coli chromosomal promoter regions in vivo (Vvedenskaya et al. 2016).

We applied these three HTS applications to comprehensively characterize a series of RNAP activities in various DNA sequence contexts.

We applied MASTER to analyze the mechanism and promoter-sequence determinants of transcription initiation. Specifically, we (1) performed a comprehensive analysis of the DNA-sequence determinants for transcription start site (TSS) selection, (2) performed a comprehensive analysis of DNA-sequence determinants for reiterative initiation (a non-standard form of transcription initiation also termed “slippage synthesis”), (3) performed a comprehensive analysis of the relationship between TSS-region sequence and transcript yield, and (4) defined the effects of DNA topology and alterations in nucleoside triphosphate (NTP) concentrations on TSS-selection, slippage synthesis, and transcript yield.

We applied CapZyme-seq to comprehensively define the promoter-sequence determinants for NCIN capping with NAD^+ in vitro and in vivo on 4^7 distinct plasmid DNA template sequence variants (Vvedenskaya et al. 2018). We also applied Capzyme-seq on E. coli chromosomal promoter regions to identify and quantify NCIN-capped small RNAs (sRNAs) in vivo.
We applied mNET-seq to determine the effects of sequence specific interactions between RNAP and core recognition element (CRE) on TSS selection. CRE is the downstream part of the nontemplate strand of the transcription bubble, and the DNA single-stranded transcription bubble is formed by RNAP unwinding ~13 bp of promoter DNA. The unwound promoter DNA and RNAP forms the open promoter complex (RP\textsubscript{O}) (Haugen, Ross, and Gourse 2008). RNAP core enzyme makes sequence-specific protein–DNA interactions with the CRE, which is mediated by the direct H-bonded interactions between the residue D446 of RNAP β subunit and the G at extreme downstream end of the transcription bubble (G\textsubscript{CRE}) (Yu Zhang et al. 2012). Using mNET-seq, we comprehensively sequenced the 5’ end of RNA transcripts generated from all chromosomal promoters in \textit{E. coli} by wild type RNAP and a mutant RNAP carrying D446A mutation in β subunit, and the D446A mutation is able to disrupt the sequence specific RNAP-CRE interaction (Vvedenskaya et al. 2016).

In addition to comprehensive profilings, we further elucidated the mechanism of transcription initiation by showing that RNAP scrunches DNA prior to RNA synthesis for TSS selection. Using MASTER and unnatural amino acid mediated protein-DNA cross-linking, we have determined the TSS, the RNAP leading-edge position, and the RNAP trailing-edge position for a library of \(4^{10}\) promoter sequences. We find that a promoter element upstream of the TSS, the “discriminator”, participates in TSS selection, and that, as the TSS changes, the RNAP leading-edge position changes, but the RNAP trailing-edge position does not change. Changes in the RNAP leading-edge position, but not the RNAP trailing-edge position, are a defining hallmark of the “DNA scrunching” that occurs concurrent with RNA synthesis in initial transcription (Winkelman and Gourse 2017). We propose that TSS selection involves DNA scrunching prior to RNA synthesis.
3.3 Methods

3.3.1 Construction of MASTER Plasmid Libraries

To generate library pMASTER-\textit{lacCONS}-N7, a PCR product containing the \textit{lacCONS} promoter with a 7-nt randomized region positioned 4–10 nt downstream of the −10 element and a 15-nt randomized "barcode" region positioned 27–41 nt downstream of the −10 element was subcloned into pSG289 (Vvedenskaya et al. 2015). Sequencing analysis of the pMASTER-\textit{lacCONS}-N7 library indicated that it contains 16,295 (~99.5%) of a possible 16,384 sequences in the TSS region. Library pMASTER-\textit{lacCONS}-N10 was generated using a similar procedure and contains 1,019,505 (~97.2%) of a possible 1,048,576 sequences carrying a 10-bp randomized region.

3.3.2 Generation of RNAs from the pMASTER-\textit{lacCONS}-N7 Library

\textit{In Vitro}

A total of 10 nM of template DNA was mixed with 50 nM RNA polymerase (RNAP) holoenzyme in transcription buffer (50 mM Tris HCl [pH 8.0], 10 mM MgCl\(_2\), 0.01 mg/ml BSA, 100 mM KCl, 5% glycerol, 10 mM DTT, 0.4 U/\mu l RNase OUT) and incubated at 37°C for 10 min. Transcription was initiated by adding NTPs (to a final concentration of 10 mM, 1 mM, or 0.1 mM each NTP) and heparin (to 0.1 mg/ml). (We note that the Mg\(^{2+}\) concentration of 10 mM is limiting. Accordingly, the effective NTP:Mg\(^{2+}\) concentration in reactions performed in the presence of 10 mM NTPs is 2.5 mM for each NTP.) Reactions were stopped after 15 min by adding EDTA to 10 mM. RNAs were analyzed by 5’ RNA-seq.
3.3.3 Generation of RNAs from the pMASTER-\textit{lacCONS-N7} Library

\textbf{In Vivo}

DH10B-T1\textsuperscript{R} cells carrying pMASTER-\textit{lacCONS-N7} were grown at 37°C in 50 ml of LB containing chloramphenicol (25 μg/ml) in a 250 ml flask shaken at 210 RPM. RNA was isolated from cells grown to an OD\textsubscript{600} ∼0.5 and used for 5’ RNA-seq analysis. Plasmid DNA was also isolated from cells and used as a template in emulsion PCR reactions to generate a product that was sequenced to assign barcodes.

3.3.4 High-Throughput Sequencing of RNA 5’ Ends (5’ RNA-Seq)

5’ RNA-seq was done using procedures described in a protocol developed by Vvedenskaya \textit{et al}. (Vvedenskaya, Goldman, and Nickels 2015). cDNA libraries were sequenced using an Illumina HiSeq 2500.

3.3.5 5’ mNET-seq

For the in vivo MASTER experiments shown in Figure 3.6A and Figure 3.6B, \textit{E. coli} DH10B-T1\textsuperscript{R} cells (Life Technologies) containing plasmids pRL706-β\textsuperscript{WT;3xFLAG} or pRL706-β\textsuperscript{D446A;3xFLAG} were transformed with ∼50 ng pMASTER-\textit{lacCONS-N7} library to obtain a 25-mL overnight culture representing cells derived from at least 20 million unique transformants; 0.5 mL of the overnight cell culture was used to inoculate 50 mL LB media containing 100 μg/μL carbenicillin and 25 μg/μL chloramphenicol. When the cell density reached an OD600 ∼0.3, 1 mM isopropyl β-D-thiogalactopyranoside (IPTG) was added, and the cells were grown for an additional 2 h. RNA associated with RNAP was isolated using procedures described in (Vvedenskaya et al. 2014).
For the experiments shown in Figure 3.6C, MG1655 cells containing plasmids pRL706-β\textsuperscript{WT:3xFLAG} or pRL706-β\textsuperscript{D446A:3xFLAG} were shaken at 220 rpm at 37 °C in 100 mL 4xLB (40 g Bacto tryptone, 20 g Bacto yeast extract, and 10 g NaCl per liter) containing 200 µg/µL carbenicillin in 500-mL DeLong flasks (Bellco). When cell density reached an OD\textsubscript{600} ~0.6, 1 mM IPTG was added, and the cells were grown for an additional 4 h. RNA associated with RNAP was isolated using procedures described in (Vvedenskaya et al. 2014).

RNA products associated with RNAP were analyzed by 5' RNA-seq using procedures described in (Vvedenskaya, Goldman, and Nickels 2015).

3.3.6 CapZyme-Seq

Isolation of RNA Products Generated In Vitro

The pMASTER-\textit{lacCONS}-N7 plasmid library (Vvedenskaya et al. 2015; Winkelman et al. 2016) contains the promoter cassette fused to the tR2 terminator. A linear DNA fragment containing the \textit{placCONS-N7} promoter library was used as a template for in vitro transcription assays. To generate this template, the plasmid pMASTER-\textit{lacCONS-N7} library variant described in (Winkelman et al. 2016) was diluted to ~109 molecules/µL. 1 µL of diluted DNA was amplified by emulsion PCR (ePCR) using a Micellula DNA Emulsion and Purification Kit in detergent-free Phusion HF reaction buffer containing 5 µg/mL BSA, 0.4 mM dNTPs, 0.5 µM Illumina RP1 primer, 0.5 µM Illumina RPI1 primer and 0.04 U/µL Phusion HF polymerase. ePCR reactions were performed with an initial denaturation step of 10 s at 95°C, amplification for 30 cycles (denaturation for 5 s at 95°C, annealing for 5 s at 60°C and extension for 15 s at 72°C), and a final extension for 5 min at 72°C. DNA was purified from emulsions according to the manufacturer’s
recommendations, recovered by ethanol precipitation, and resuspended in nuclease-free water to a final concentration of ~1 μM.

The sequence of the linear DNA template used in the transcription assays is:

5′-gttcagttctacagtccagatcagggcTTGACActttatgctcggctcgTATAATgtgNNNNNNNgtagcggataacatNNNNNNNNNNNNNcctgcaggtggagttctcgggtgccaaggaactccagttcacacgtc-3′, where the −35 and −10 elements are in bold, and the 7-bp randomized transcription start site region and 15-bp transcribed-region barcode are underlined.

In vitro transcription was performed by mixing 10 nM of template DNA with 50 nM RNAP holoenzyme in 50 mM Tris HCl (pH 8.0), 10 mM MgCl₂, 0.01 mg/ml BSA, 100 mM KCl, 5% glycerol, 10 mM DTT, and 0.4U/μl RNaseOUT. Reactions were incubated at 37°C for 10 min to form open complexes. A single round of transcription was initiated by addition of 100 μM ATP, 100 μM CTP, 100 μM UTP, 100 μM GTP, and 0.1 mg/ml heparin, or by the addition of 100 μM ATP, 100 μM CTP, 100 μM UTP, 100 μM GTP, 2 mM NAD⁺, and 0.1 mg/ml heparin. Reactions were incubated at 37°C for 15 min and stopped by addition of 0.5M EDTA (pH 8) to a final concentration of 50 mM. Nucleic acids were recovered by ethanol precipitation and resuspended in 30 μL of nuclease-free water. Reactions performed in the absence or presence of NAD⁺ were performed in triplicate, and each replicate was analyzed by both CapZyme-seq^{NudC} and CapZyme-seq^{Rai1}.

The resuspended nucleic acids were mixed with 30 μL of 2x RNA loading dye and separated by electrophoresis on 10% 7M urea slab gels (equilibrated and run in 1 x TBE). The gel was stained with SYBR Gold nucleic acid gel stain, bands were visualized
on a UV transilluminator, and RNA products ~100 nt in size were excised from the gel. The excised gel slice was crushed, 300 μL of 0.3 M NaCl in 1x TE buffer was added, and the mixture was incubated at 70°C for 10 min. Eluted RNAs were collected using a Spin-X column. After the first elution, the crushed gel fragments were collected and the elution procedure was repeated, nucleic acids were collected, pooled with the first elution, isolated by ethanol precipitation, and resuspended in 10 μL of RNase-free water.

**Isolation of RNAs Generated In Vivo in *E. coli***

To analyze RNA products generated from the pMASTER-lacCONS-N7 library in vivo, three independent 50 mL cell cultures of DH10B-T1R cells containing the pMASTER-lacCONS-N7 plasmid library variant described in (Vvedenskaya et al. 2015) were grown in LB media containing chloramphenicol (25 μg/ml) in a 250 mL DeLong flask shaken at 210 RPM at 37°C to mid-exponential phase (OD600 ~0.5). 2 mL aliquots of the cell suspensions were placed in 2 mL tubes and cells were collected by centrifugation (1 min, 21,000 x g at room temperature). Supernatants were removed and cell pellets were rapidly frozen on dry ice and stored at −80°C.

RNA was isolated from each cell pellet as in (Vvedenskaya et al. 2015). Cell pellets were resuspended in 600 μL of TRI Reagent solution. The suspensions were incubated at 70°C for 10 min and centrifuged (10 min, 21,000 x g) to remove insoluble material. The supernatant was transferred to a fresh tube, ethanol was added to a final concentration of 60.5%, and the mixture was applied to a Direct-zol spin column. DNase I treatment was performed on-column according to the manufacturer’s recommendations. RNA was eluted from the column using nuclease-free water that had been heated to 70°C (3 × 30 μL elutions; total volume of eluate = 90 μl). RNA was treated with 2U TURBO DNase at 37°C for 1 h to remove residual DNA. Samples were extracted with acid
phenol:chloroform, RNA was recovered by ethanol precipitation and resuspended in RNase-free water. A MICROBExpress Kit was used to deplete rRNAs from 9 μg of the recovered RNA. The rRNA-depleted RNA was isolated by ethanol precipitation and resuspended in 30 μL of RNase-free water.

pMASTER-\textit{lacCONS-N7} plasmid DNA was isolated from each of the three cultures using a Plasmid Mini-prep kit. Plasmid DNA isolated from these cultures was used as template in ePCR reactions to generate linear DNA products that were analyzed by high-throughput sequencing.

The sequence from the −35 element of the lacCONS-N7 promoter to the end of the tR2 terminator on the plasmid used for the in vivo experiments is:

\begin{verbatim}
5'-
TTGACA CTTTATGCTTGGCTC GTATA ATGTGNNNNNNNGTGAGCGGATAACAAATN
NNNNNNNNNNNNTGGAATTCTCGGGTGCCAAGGCCCAGCGGCCGTCTTCAA
GAGCTCATGGATCCGAATAGCCATCCCAATCGAACAGGCCTGCTGGTAATCGCAG
GCCTTTTTATTTGGAT-3', where the −35 and −10 elements are in bold, and the 7-bp randomized transcription start site region and 15-bp transcribed-region barcode are underlined. Note that the pMASTER-\textit{lacCONS-N7} library variant described in (Winkelman et al. 2016) used to generate the linear DNA template for in vitro assays contains an 8-bp sequence insertion (CCTGCAGG) downstream of the transcribed-region barcode that is not present in the pMASTER-\textit{lacCONS-N7} plasmid library variant described in (Vvedenskaya et al. 2015) used for the in vivo assays.
\end{verbatim}

Enzymatic Treatments of RNA Products Generated In Vitro with Rpp, NudC, or Rai1
To convert 5’ triphosphate RNA to 5’ monophosphate RNA, products were mixed with 20U Rpp and 40U RNaseOUT in 1x Rpp reaction buffer in a 20 μL reaction volume.

To convert NAD$^+$-capped RNA to 5’ monophosphate RNA, products were mixed with 1x NudC reaction buffer, 3.6 μM NudC, and 40U of RNaseOUT in a 20 μL reaction volume or with 1x Rai1 reaction buffer, 0.3 μM Rai1, and 40U RNaseOUT in a 20 μL reaction volume. In parallel, we added RNA products to each of the reaction buffers without addition of enzyme. Reactions were incubated at 37°C for 30 min, 20 μL of 2x RNA loading dye was added, and the products were separated by electrophoresis on 10% 7M urea slab gels (equilibrated and run in 1x TBE). The gel was stained with SYBR Gold nucleic acid gel stain, bands were visualized on a UV transilluminator, and ~100-nt products were excised from the gel. The excised gel slice was crushed, 300 μL of 0.3 M NaCl in 1xTE buffer was added, and the mixture was incubated at 70°C for 10 min. The eluate was collected using a Spin-X column. After the first elution step, the elution procedure was repeated, eluates were pooled, and RNA was isolated by ethanol precipitation and resuspended in 10 μL of RNase-free water.

**Enzymatic Treatments of RNA Products Generated In Vivo with Rpp or NudC**

To convert 5’-triphosphate RNA to 5’-monophosphate RNA, 2 μg rRNA-depleted RNA were mixed with 20U Rpp and 40U RNaseOUT in 1x Rpp reaction buffer in a 30 μL reaction volume. In parallel, we added 2 μg rRNA-depleted RNA to 1x Rpp reaction buffer without addition of enzyme. Reactions were incubated at 37°C for 30 min. Samples were extracted with acid phenol:chloroform, RNA was recovered by ethanol precipitation, and resuspended in 10 μL RNase-free water.
Prior to treating total cellular RNA with NudC to convert NCIN-capped RNA to 5’-monophosphate RNA we first treated 2 μg of rRNA-depleted RNA with 2U CIP in the presence of 40U RNaseOUT in a 30 μL reaction volume at 37°C for 1h to remove 5’-terminal phosphates. RNA was extracted with acid phenol:chloroform, recovered by ethanol precipitation, and resuspended in 20 μL of RNase-free water. CIP-treated RNA was mixed with 3.75 μM NudC and 40U RNaseOUT in 1x NudC reaction buffer in a 30 μL reaction volume. In parallel, CIP-treated RNA was incubated in 1x NudC reaction buffer without NudC. Reactions were incubated at 37°C for 30 min, 30 μL of 2x RNA loading dye was added, and the products were separated by electrophoresis on 10% 7M urea slab gels (equilibrated and run in 1x TBE). The gel was stained with SYBR Gold nucleic acid gel stain, RNA was visualized on a UV transilluminator, and excised from the gel. The excised gel slice was crushed, 400 μL of 0.3 M NaCl in 1x TE buffer was added, and the mixture was incubated at 70°C for 10 min. The eluate was collected using a Spin-X column. After the first elution step, the elution procedure was repeated, eluates were pooled, and recovered RNA isolated by ethanol precipitation and resuspended in 10 μL of RNase-free water.

5’ Adaptor Ligation

RNA products (in 10 μL of nuclease-free water) were combined with PEG 8000 (10% final concentration), 1 mM ATP, 40U RNaseOUT, 1x T4 RNA ligase buffer, and 10 U T4 RNA ligase 1, in 30 μL reaction volume. 0.3 μM barcoded 5’ adaptor oligo was added to in vitro generated RNAs, and 1 μM barcoded 5’ adaptor oligo was added to in vivo generated RNAs, respectively. Reactions were incubated at 16°C for 16 h.

To enable quantitative comparisons between samples treated with Rpp, samples treated with NCIN-processing enzymes, samples incubated in Rpp reaction buffer (“mock Rpp
treatment"), and samples incubated in NCIN-processing enzyme reaction buffer ("mock NudC treatment" or "mock Rai1 treatment"), we performed the 5′ adaptor ligation step using barcoded 5′-adaptor oligonucleotides. For libraries prepared from RNA generated in vitro or in vivo, oligo i105 was used in ligation reactions performed with products isolated after Rpp treatment, oligo i106 was used in ligation reactions performed with products isolated after mock Rpp treatment, oligo i107 was used in ligation reactions performed with products isolated after NudC treatment (for CapZyme-seq<sup>NudC</sup>) or Rai1 treatment (for CapZyme-seq<sup>Rai1</sup>), and oligo i108 was used in ligation reactions performed with products isolated after mock NudC treatment or mock Rai1 treatment.

The ligation reactions were stopped by adding 30 μL of 2x RNA loading dye and heated at 95°C for 3 min. Each set of four adaptor ligation reactions were combined, mixed with an equal volume of 2x RNA loading dye, and separated by electrophoresis on 10% 7M urea slab gels (equilibrated and run in 1x TBE). Gels were incubated with SYBR Gold nucleic acid gel stain, and bands were visualized with UV transillumination. For RNA generated in vitro, ~150 nt products were recovered from the gel and resuspended in 10 μL of nuclease-free water. For RNA generated in vivo, products migrating above the 5′-adaptor oligo were recovered from the gel and resuspended in 50 μL of nuclease-free water.

First Strand cDNA Synthesis: Analysis of RNAs Generated from the lacCONS-N7 Promoter Library In Vitro

5′-adaptor-ligated products (in 10 μL of nuclease-free water) were mixed with 1.5 μM s128A oligonucleotide, incubated at 65°C for 5 min, then cooled to 4°C. To this mixture was added 9.7 μL of a solution containing 4 μL of 5x First-Strand buffer, 1 μL of 10 mM dNTP mix, 1 μL of 100 mM DTT, 1 μL (40U) RNaseOUT, 1 μL (200U) of SuperScript III
Reverse Transcriptase and 1.7 μL of nuclease-free water. Reactions were incubated at 25°C for 5 min, 55°C for 60 min, 70°C for 15 min, then cooled to 25°C. 10U of RNase H was added, the reactions were incubated at 37°C for 20 min and 20 μL of 2X RNA loading dye was added. Nucleic acids were separated by electrophoresis on 10% 7M urea slab gels (equilibrated and run in 1X TBE). Gels were incubated with SYBR Gold nucleic acid gel stain, bands were visualized with UV transillumination, and species ~80 to ~150 nt in length were recovered from the gel and resuspended in 20 μL of nuclease-free water.

First Strand cDNA Synthesis: Analysis of RNAs Generated In Vivo

5’-adaptor-ligated products were divided into two equal portions (each in 25 μL of nuclease-free water). One portion was mixed with 0.5 μL of 100 μM s128A oligonucleotide to enable analysis of RNA products generated from the lacCONS-N7 promoter library, while the other portion was mixed with 0.5 μL of 100 μM sRNA oligo pool (a mixture of 77 oligonucleotides each having a 3’ end sequence complementary to positions +50 to +30 of one of 77 annotated sRNAs in E. coli; see Key Resources Table). The mixtures were incubated at 65°C for 5 min, then cooled to 4°C. To these mixtures was added 24.5 μL of a solution containing 10 μL 5X First-Strand buffer, 2.5 μL 10 mM dNTP mix, 2.5 μL 100 mM DTT, 2.5 μL (100U) RNaseOUT, 2.5 μL (500U) SuperScript III Reverse Transcriptase and 4.5 μL of nuclease-free water. Reactions were incubated at 25°C for 5 min, 55°C for 60 min, 70°C for 15 min, then cooled to 25°C. 20U RNase H was added, the reactions were incubated at 37°C for 20 min and 50 μL of 2X RNA loading dye was added. Nucleic acids were separated by electrophoresis on 10% 7M urea slab gels. To isolate cDNA derived from the lacCONS-N7 promoter library, ~80- to ~150-nt products were recovered from the gel (procedure as above) and resuspended in 20 μL of nuclease-free water. To isolate cDNA derived from sRNA, ~80-
to ~225-nt products were recovered from the gel and resuspended in 20 μL of nuclease-free water.

**cDNA Amplification**

cDNA derived from RNA products generated in vitro or in vivo were diluted with nuclease-free water to a concentration of ~10^9 molecules/μl. 2 μL of the diluted cDNA solution was used as a template for ePCR reactions containing Illumina index primers using a Micellula DNA Emulsion and Purification Kit (20 cycles; conditions as above). The emulsion was broken and DNA was purified according to the manufacturer’s recommendations. DNA was recovered by ethanol precipitation and resuspended in 20 μL of nuclease-free water.

**High-Throughput Sequencing**

Barcoded libraries were pooled and sequenced on an Illumina NextSeq platform in high-output mode using custom primer s1115.

**3.3.7 MASTER-N10-XL: crosslinking of Bpa-containing RNAP in RPo**

50 μl reactions containing 10 nM pMASTER-λacCONS-N10 plasmid DNA and 50 nM Bpa-containing RNAP in transcription buffer were incubated at 37° for 15 min before irradiating with 365 nM UV-light for 10 total min (2 min of UV-irradiation followed by 1 min incubation at 37°, repeated five times).
3.3.8 MASTER-N10-XL: primer extension mapping of RNAP-DNA crosslinks in RPo

Primer extension reactions were performed in a total volume of 50 μl by combining 8 μl of each crosslinking reaction with 5 units of Taq DNA polymerase (NEB), 5 μl 10X Taq buffer (NEB), 250 μM of each dNTP (NEB), 2M betaine, 5% DMSO, and 10 pmol of primer 7210. Primer 7210 anneals to the non-template-strand of the plasmid backbone, 50-70 nt downstream from the -10 hexamer. Extension products were amplified by repeating the following steps for 18 cycles: 30 s at 95°C, 30 s at 53°, and 30 s at 72°.

3.3.9 MASTER-N10-XL: isolation of primer extension products

Primer extension products were recovered after overnight precipitation, mixed with an equal volume of 2x RNA loading dye (95% deionized formamide, 18 mM EDTA, 0.25% SDS, xylene cyanol, bromophenol blue, amaranth), and separated by electrophoresis on 10% 7M urea slab gels (equilibrated and run in 1x TBE). The gel was stained with SYBR Gold nucleic acid gel stain (Life Technologies) and DNA ~50- to ~120-nt in size (in the case of experiments with RNAP-β'T48Bpa) and ~30- to ~90-nt in size (in the case of experiments with RNAP-β'R1148Bpa) were excised from the gel. The isolated gel fragments were crushed, incubated in 300 μl of 0.3 M NaCl in 1x TE buffer at 70°C for 10 min, and the supernatant was collected using a Spin-X column (Corning). The elution procedure was repeated once, supernatants were combined, nucleic acids were recovered by ethanol precipitation, washed twice with 80% cold ethanol, and resuspended in 5 μl of nuclease-free water.
3.3.10 MASTER-N10-XL: ligation of adaptor to 3’ end of primer extension products

The 5 μl solution of the extension products was combined with 1 μl 10X NEB buffer 1, ~0.8 μM Illumina-specific s1248 oligo (5’ adenylated and 3’-end blocked oligo with 10N randomized sequence on the 5’ end), 5 mM MnCl2 and 1 μM of 5’AppDNA/RNA ligase (NEB) in a final volume of 10 μl. The mixture was incubated for 1 h at 65°C followed by 3 min at 90°C, and cooled to 4°C for 5 min. The reaction mixture was then combined with 15 μl of T4 RNA ligase 1 cocktail containing 1x T4 RNA ligase 1 reaction buffer (NEB), 12% PEG 8000, 10 mM DTT, 60 μg/mL BSA and 10 U of T4 RNA ligase 1 (NEB). The mixture was incubated at 16°C for 16 h.

3.3.11 MASTER-N10-XL: size selection of adaptor-ligated DNA products

Adaptor-ligated products were separated by electrophoresis on 10% 7M urea slab gels (equilibrated and run in 1x TBE). The gel was stained with SYBR Gold nucleic acid gel stain and species ranging from ~90- to ~140-nt (RNAP-β’T48Bpa) or ~75- to ~120-nt (RNAP-β’R1148Bpa) were excised from the gel. DNA was eluted from gel, isolated by ethanol precipitation and resuspended in 10 μl of nuclease-free water.

3.3.12 MASTER-N10-XL: amplification of DNAs

5 μl of the adaptor ligated DNA products were added to a 35 μl mixture containing 1x Phusion HF reaction buffer, 0.2 mM dNTPs, 0.25 μM Illumina RP1 primer, 0.25 μM Illumina RPI index primer and 0.02 U/μl Phusion HF polymerase. PCR was performed with an initial denaturation step of 30 s at 98°C, amplification for 11 cycles (denaturation
for 10 s at 98°C, annealing for 20 s at 62°C, and extension for 10 s at 72°C), and a final extension for 5 min at 72°C.

3.3.13 MASTER-N10-XL: size selection of libraries

After amplification, DNAs were separated by electrophoresis on a non-denaturing 10% slab gel (equilibrated and run in 1 X TBE), and DNAs between ~180 bp and ~190 bp (RNAP-β'T48Bpa) and between ~160 bp and ~170 bp (RNAP-β'R1148Bpa) were isolated by gel excision. Amplified DNAs were eluted from the gel with 600 μL of 0.3M NaCl in 1xTE buffer at 37°C for 2 h, precipitated and resuspended in 20 μL of nuclease free water.

3.3.14 MASTER and MASTER-XL: high-throughput sequencing

Barcoded libraries were sequenced on the HiSeq 2500 platform in rapid mode using a custom sequencing primer s1115.

3.3.15 MASTER Data Analysis

Sequencing of template DNA was used to associate the 7-bp randomized sequence in the region of interest with a corresponding second 15-bp randomized sequence that serves as its barcode. For 5’ RNA-seq analysis, we considered only those reads that contained a perfect match to the sequence downstream of position 10 and a perfect match to the five bases downstream of the 15-bp barcode. The identity of the 15-bp barcode was used to determine the identity of bases at positions 4–10 of the lacCONS template. Sequences derived from the RNA 5’ end of reads that were perfect matches to the sequence of the template were used for analysis of TSS selection. Sequences derived from RNA 5’ ends that carried one or more mismatches from the DNA template
and/or extended up to five bases upstream of position 4 were not considered in the analysis of TSS position, but were used for analysis of productive slippage synthesis and for analysis of transcript yields.

Source code and documentation for analysis of DNA templates and 5’ RNA-seq libraries are provided at [https://github.com/NickelsLabRutgers/MASTER-Data-Analysis](https://github.com/NickelsLabRutgers/MASTER-Data-Analysis).

### 3.3.16 5’ mNET-seq Data Analysis

Identification of TSS positions and TSS regions for natural promoters in *E. coli* was done essentially as described in (Druzhinin et al. 2015). The first six bases of each read were trimmed (to remove sequences introduced during the cDNA library construction procedure), and the next 30 bases were aligned to the *E. coli* reference genome (NC_000913.3) using Bowtie (Langmead et al. 2009). Among these reads, we used those that aligned to a unique position in the genome with zero mismatches for the analysis of TSS selection.

Using data derived from the analysis of RNA products associated with RNAP-βWT, we defined a list of primary TSS positions that meet the following two criteria: (i) the read count at the coordinate was above a threshold value (≥ 50 reads) and (ii) the read count at the coordinate represented a local maximum in an 11-bp window centered on the coordinate. For each primary TSS position, we designated the positions spanning 5-bp upstream to 5-bp downstream as a TSS region. Next, for each TSS region, we calculated the percentage of reads starting at each of the 11 positions: \( %TSS_Y = 100 \times \frac{(#\text{reads starting at position } Y)}{\text{total } #\text{ reads starting within the TSS region}} \). We identified 1,500 TSS positions within TSS regions with an above-threshold value of %TSS (≥ 20%). For each of these 1,500 TSS positions, we calculated the difference between the
average %TSS observed in experiments performed with RNAP-\(\beta^{\text{WT}}\) and that observed in experiments performed with RNAP-\(\beta^{\text{D446A}}\).

3.3.17 CapZyme-Seq Data Analysis

Transcribed-Region Barcode Identification for MASTER-\(\text{lacCONS-N7}\) Experiments

The DNA template used for in vitro transcription reactions was analyzed by high-throughput sequencing to identify transcribed-region barcodes, and to assign these barcodes to individual \(\text{placCONS-N7}\) template sequences (Vvedenskaya et al. 2015).

For in vivo MASTER-\(\text{lacCONS-N7}\) experiments, DNA products generated in ePCR reactions using pMASTER-\(\text{lacCONS-N7}\) plasmid DNA isolated from cells were analyzed by high-throughput sequencing to identify transcribed-region barcodes, and to assign these barcodes to individual \(\text{placCONS}\) template sequences (Vvedenskaya et al. 2015).

Analysis of Sequencing Reads Derived from cDNA

All cDNA libraries were generated from the same input RNA that had been split into four portions and subjected to two distinct 5′ processing reactions (Rpp and NudC or Rai1) and two distinct control reactions (mock Rpp treatment and mock NudC or Rai1 treatment). To distinguish cDNA derived from each of the four reactions, RNA from each reaction was ligated to a 5′-adaptor oligonucleotide containing a unique 4-nt barcode sequence (i105, i106, i107, or i108; see above). Each 5′-adaptor oligonucleotide also contains 11 nt of random sequence at the 3′ end that improves ligation efficiency (by reducing sequence-dependent effects), and that marks individual RNA products with an 11-nt sequence tag that is used to reduce the effects of PCR amplification bias. Thus, sequencing reads having identical 11-nt sequence tags and identical cDNA insert sequences are counted as a single read count during the data analysis.
Due to the presence of the 4-nt barcode sequence and 11-nt sequence tag at the 3’ end of the 5’-adaptor oligonucleotide, the first four bases of each read provide the sequence of the 4-nt barcode, the next 11 bases provide the sequence of the 11-nt sequence tag, and the 16th base provides the sequence of the RNA 5’ end from which the cDNA was generated.

Analysis of Sequencing Reads Derived from cDNA: MASTER-lacCONS-N7

Analysis In Vitro

For analysis of cDNA libraries generated from RNA products produced from the MASTER-lacCONS-N7 library in vitro the 15-base transcribed-region barcode was identified and used to associate reads derived from RNA transcripts with their template of origin (Vvedenskaya et al. 2015). We considered only RNA 5’ end-sequences that could be aligned to the sequence of their template of origin with no mismatches. These reads were associated with one of the four reaction conditions based on the identity of the 4-nt barcode sequence.

For each of the ~16,000 sequence variants, we determined the number of reads emanating from each position of the N7 region for samples treated with Rpp (#Rpp), samples treated with NCIN-processing enzymes (#NudC or #Rai1), samples subjected to mock Rpp treatment (#Rppmock), and samples subjected to mock NCIN-processing enzyme treatment (NudCmock or #Rai1mock). From these values, we calculated #ppp and #NCIN, where #ppp = (#Rpp - #Rppmock), and #NCIN = (#NudC - #NudCmock) for CapZyme-seqNudC and #NCIN = (#Rai1 - #Rai1mock) for CapZyme-seqRai1. Negative values for #ppp or #NCIN were replaced with a value of 0.
Analysis of reactions performed in the absence of NAD$^+$ revealed activity of NudC and Rai1 on 5′-triphosphate RNA. By comparison with analysis of reactions performed in the absence of NAD$^+$ with Rpp we estimate, on average, NudC converted ~19% of 5′-triphosphate RNA to 5′-monophosphate RNA, and Rai1 converted ~3.5% of 5′-triphosphate RNA to 5′-monophosphate RNA. Therefore, to account for the conversion of 5′-triphosphate RNA to 5′-monophosphate RNA by NudC or Rai1 we used values of #ppp and #NCIN obtained in reactions performed in the absence of NAD$^+$ to calculate a correction factor (cf.), where cf. = (#NCIN / #ppp), to apply to the analysis of reactions performed in the presence of NAD$^+$.

To analyze results of reactions performed in the presence of NAD$^+$ we used the value of cf., the values for #ppp, and the value for #NCIN to calculate a "background corrected" value of #NCIN (#NCIN$^{Bkd\_C}_{Cor}$), where #NCIN$^{Bkd\_C}_{Cor}$ = [#NCIN – (cf. x #ppp)]. Next, using the value of #NCIN$^{Bkd\_C}_{Cor}$ and #ppp we calculated a value of "percent capping," where percent capping = 100% x [#NCIN$^{Bkd\_C}_{Cor}$ / (#NCIN$^{Bkd\_C}_{Cor}$ + #ppp)]. (Note that the values of (#NCIN$^{Bkd\_C}_{Cor}$ + #ppp) of 0 were replaced by “1” prior to calculating percent capping.)

For results of experiments performed in vitro we calculated a value for “capping efficiency,” where capping efficiency = [percent capping / (100 - percent capping)] / 20. (Note that the value of 20 corresponds to the [NAD$^+$]/[ATP] for the in vitro transcription experiments performed in this work.

Results of Figure 3.7C represent a plot of the mean percent capping value (n = 3; replicates of CapZyme-seq$^{NudC}$ or CapZyme-seq$^{Rai1}$) for template sequences with (#NCIN$^{Bkd\_C}_{Cor}$ + #ppp) ≥ 50.
Results of Figure 3.8B represent the mean TSS value (n = 3; replicates of CapZyme-seq^{NudC} or CapZyme-seq^{Rai1}) and mean %TSS values (n = 6; replicates of CapZyme-seq^{NudC} and CapZyme-seq^{Rai1}) for template sequences with #NCIN^{Bkd_Cor} ≥ 50. Results of Figure 3.8C represent the mean TSS value (n = 3; replicates of CapZyme-seq^{NudC} or CapZyme-seq^{Rai1}) and mean %TSS values (n = 6; replicates of CapZyme-seq^{NudC} and CapZyme-seq^{Rai1}) for template sequences with #ppp ≥ 50. Mean TSS and %TSS were calculated using the formulas below.

\[
\%TSS_X = 100 \times \frac{\#\text{reads at position } X}{\#\text{reads at position 4-10}}
\]

\[
\text{mean TSS} = \frac{[4 \times \%TSS \text{ at position 4}] + [5 \times \%TSS \text{ at position 5}] + [6 \times \%TSS \text{ at position 6}] + [7 \times \%TSS \text{ at position 7}] + [8 \times \%TSS \text{ at position 8}] + [9 \times \%TSS \text{ at position 9}] + [10 \times \%TSS \text{ at position 10}]}{100}
\]

Results of Figure 3.9 and Figure 3.10B represent relative capping efficiency values calculated for the position 7 bp downstream of the promoter −10 element for template sequences with (#NCIN^{Bkd_Cor} at position 7 + #ppp at position 7) ≥ 25 in each of the three CapZyme-seq^{NudC} replicates and in each of the three CapZyme-seq^{Rai1} replicates. Relative capping efficiency at each position was determined for groups of four promoter sequences, “quartets,” having A, G, C, or T, along with sequences identical at each other position. The relative capping efficiency of each A+1 promoter sequence, Y, at each position, X, is calculated by dividing the capping efficiency of Y by the mean capping efficiency of Y’s quartet at position X. For each promoter sequence, the mean capping efficiency (n = 6; replicates of CapZyme-seq^{NudC} and CapZyme-seq^{Rai1}) was used to calculate relative capping efficiency at each position. For results of Figure 3.10C, the relative capping efficiency of the consensus and anti-consensus promoter sequences
were calculated by dividing the capping efficiency of each consensus or anti-consensus promoter sequence by the mean capping efficiency of all A⁺¹ promoter sequences.

Uncapped RNA products generated in vitro from A⁺¹ promoters have the sequence:

5’-
ANNNGUGAGCGGAUAACAAUNNNNNNNNNNNNNCCUGCAGGUGAAUUCUCGGGUGCCAAGGAACUCCAGUCACAUCACGAUCUCGUAGCCGUCUCUGCUUG-3’.

**Analysis of Sequencing Reads Derived from cDNA: MASTER-lacCONS-N7**

**Analysis In Vivo**

For analysis of cDNA libraries generated from RNA products produced from the MASTER-lacCONS-N7 library in vivo the 15-base transcribed-region barcode was used to associate reads derived from RNA transcripts with their template of origin (Vvedenskaya et al. 2015). As with RNAs generated in vitro, we also considered only RNA 5’ end-sequences that could be aligned to the sequence of their template of origin with no mismatches (Vvedenskaya et al. 2015). These reads were associated with one of the four reaction conditions based on the identity of the 4-nt barcode sequence.

For each of the ~16,000 sequence variants, we determined the number of reads emanating from each position of the N7 region for samples treated with Rpp (#Rpp), samples subjected to mock Rpp treatment (#Rppmock), samples subjected to NudC treatment (#NudC), and samples subjected to mock NudC treatment (NudCmock). Using the values for #NudC and #NudCmock we calculated the value of #NCIN, where #NCIN = (#NudC - #NudCmock). Negative values for #NCIN were replaced with a value of 0.
To avoid complications due to the background activity of NudC on 5′-triphosphate RNA we treated RNA generated in vivo with CIP to remove 5′-terminal phosphates prior to treatment with NudC. In contrast to the analysis of RNA produced in vitro, removal of 5′-terminal phosphates by CIP prior to NudC treatment enabled us to directly use the value of #NCIN to calculate percent capping for RNA produced in vivo.

To calculate percent capping in vivo we used the value for #Rpp instead of that for #ppp in order to include 5′-monophosphate RNA in our analysis. Thus, we calculated a value of percent capping in vivo using the formula: percent capping = 100% × (#NCIN / (#NCIN + #Rpp)).

For results shown in Figure 3.11A–D, analysis was done using read count sums of the three independent CapZyme-seq\textsuperscript{NudC} replicates. Results of Figure 3.11A represent a plot of the mean percent capping values for template sequences with (#NCIN + #Rpp) ≥ 50. Results of Figure 3.11B (left) represent the mean TSS value and mean %TSS values calculated using the sum of #NCIN of all template sequences. Results of Figure 3.11B (right) represent the mean TSS value and mean %TSS values calculated using the sum of #ppp of all template sequences. %TSS and the mean TSS were calculated using the formulas below.

\[
\%\text{TSS}_x = 100 \times \frac{\text{#reads at position } X}{\text{#reads at position 4-10}}
\]

\[
\text{mean TSS} = \frac{[(4 \times \%\text{TSS at position 4}) + (5 \times \%\text{TSS at position 5}) + (6 \times \%\text{TSS at position 6}) + (7 \times \%\text{TSS at position 7}) + (8 \times \%\text{TSS at position 8}) + (9 \times \%\text{TSS at position 9}) + (10 \times \%\text{TSS at position 10})]}{100}
\]
Results of Figure 1C represent relative difference in percent capping values calculated for the position 7 bp downstream of the promoter −10 element for template sequences with (#NCIN at position 7 + #ppp at position 7) ≥ 25. Relative difference in percent capping at each position was determined for groups of four promoter sequences, “quartets,” having A, G, C, or T, along with sequences identical at each other position. The relative difference in percent capping of each A_{+1} promoter sequence, Y, at each position, X, is calculated by subtracting the percent capping of Y by the mean percent capping of Y’s quartet at position X. For results of Figure 3.11D, percent capping values of the consensus and anti-consensus promoter sequences are shown.

Uncapped RNA products generated in vivo from A+1 promoters have the sequence:

5’-
ANNNGUGAGCGGAUAACAAUNNNNNNNNNNNNNUGGAAUUCUCGGGUGCCAA
GGGCCCCAGCGGCCGUCUUCAAGAGCUCAUGGAUCCGAAUAGCCAUCCCAAUCG
AACAGGCCUGCGUGUAACGCAGGCCUUUUUAAUUUGGAU-3’

**Analysis of Sequencing Reads Derived from cDNA: sRNA Analysis**

Reads were associated with one of the four reaction conditions based on the identity of the 4-nt barcode sequence, the next 11 bases provide the sequence of the 11-nt sequence tag, and the 16th base provides the sequence of the RNA 5’ end from which the cDNA was generated. The first 20 bases of the RNA 5’ end-sequence of each sequencing read was mapped exactly to the sequences from 50 bp upstream to 50 bp downstream of the annotated 5’ end position of each sRNA. Read counts derived from samples treated with Rpp (#Rpp), samples subjected to mock Rpp treatment (#Rpp_mock), samples subjected to NudC treatment (#NudC), and the number of reads for samples subjected to mock NudC treatment (NudC_mock). Using the values for #NudC and
#NudCmock we calculated the value of #NCIN, where #NCIN = (#NudC - #NudCmock).
Negative values for #NCIN were replaced with a value of 0. As mentioned above, removal of 5’-terminal phosphates by CIP prior to NudC treatment enabled us to directly use the value of #NCIN to calculate percent capping for sRNA produced in vivo.

We first used values for #Rpp and #Rppmock to identify positions with the highest value of #Rpp for each sRNA. Next, for these positions we identified those representing primary TSS where the value of #Rpp was at least two times greater than #Rppmock. In this manner, we identified 16 primary TSS positions where the base pair is A:T and the sum of #Rpp and #NCIN is greater than 100 in each of the three replicates (Figure 3.11E).
For these primary TSS, we calculated a value of percent capping using the formula:
percent capping = 100% x [#NCIN / (#NCIN + #Rpp)]. Values reported in Figure 3.11E are the mean of three replicates.

3.3.18 MASTER and MASTER-XL: analysis of DNA templates
High-throughput sequencing of template DNA was used to associate the 10-bp or 7-bp randomized sequence with a corresponding 15-bp barcode. Barcodes present on the pMASTER-lacCONS-N10 or pMASTER-lacCONS-N7 plasmid templates were assigned as described in MASTER data analysis. In brief, plasmid DNA is used as template in an emulsion PCR reaction using Illumina RP1 primer and Illumina RPI index primer. The ePCR product generated from this procedure carries the sequence of the promoter, the sequence of the 10-bp or 7-bp randomized region of interest, and the sequence of the 15-bp index region. After sequencing the ePCR products, we identified reads that were a perfect match to the conserved regions of the MASTER-lacCONS-N7 sequence:
aggcttgacactttatatgctggtgccaagg (the -35 element and -10 element of placCONS-N7 are in bold; the 7-
bp region of interest and the 15-bp barcode region are underlined) or the MASTER-
lacCONS-N10 template:

```
aggcttgaca cttatatgcgtcgtcgtataattnnnnnnnntngtgcagccgataacaattnnnnnnnnnnntggattctcgggtgccaagg
```

(the -35 element and -10 element of placCONS-N10 are in bold; the
10-bp region of interest and the 15-bp barcode region are underlined). From these reads
we identified 15-bp barcodes that could be uniquely assigned to one of the N7 or N10
sequences in the randomized region of interest. These 15-bp barcodes were used to
associate reads derived from RNA transcripts or primer extension products with their
template of origin.

### 3.3.19 MASTER-N10 and MASTER-N7: analysis of TSS selection

Sequences derived from the RNA 5’ end of reads that were perfect matches to the
sequence of the template from which they were derived were used to identify the TSS
position as described in (2). We note that sequences derived from the RNA 5’ end begin
at the 16th bp of each read because the 5’ adaptor used to generate the cDNAs (s1206)
carries 15 randomized bases at the 3’ end to improve ligation efficiency and provide a
“digital tag” to individual transcripts. These 15-base digital tags are used to identify
cDNAs that are preferentially amplified during PCR and count these species only once
(i.e. as a single read count) during analysis. The mean TSS was determined for each of
the ~16,000 discriminator sequences using the formula below:

\[
\text{mean TSS} = (4 \times (\text{sum of the read count at TSS} = 4)) + (5 \times (\text{sum of the read count at TSS} = 5)) + (6 \times (\text{sum of the read count at TSS} = 6)) + (7 \times (\text{sum of the read count at TSS} = 7)) + (8 \times (\text{sum of the read count at TSS} = 8)) + (9 \times (\text{sum of the read count at TSS} = 9))
\]
TSS = 9) + (10 x (sum of the read count at TSS = 10)) / (sum of the read count at TSS=4 to TSS=10)

To establish whether any differences in the mean TSS values for distinct discriminator sequences were significant we compared the mean TSS values for each of the 64 discriminator sequences to the mean TSS values for each of the other 63 discriminator sequences by bootstrapping 10,000 subsets of the data for each discriminator sequence. The differences in the mean TSS for each subset were fit to a Gaussian distribution using maximum likelihood estimation, and the fitness was evaluated using Kolmogorov–Smirnov test. Using the fitted distribution, the p-value of the two discriminator sequences being compared having a significant difference in the mean TSS was calculated. The p-values of multiple comparisons were adjusted using Benjamini–Hochberg procedure.

3.3.20 MASTER-N10-XL: analysis of the positions of the RNAP trailing edge and the RNAP leading edge

Sequences derived from the 3′ ends of primer extension products that were perfect matches to the sequence of the template from which they were derived were used to define the RNAP trailing-edge and RNAP leading-edge positions using analyses similar to those used to define the TSS position. We note that sequences derived from the 3′ ends of primer extension products begin at the 11th bp of each read because the 3′ adaptor used in the library construction procedure (s1248) carries 10 randomized bases at the 5′ end to improve ligation efficiency and provide a “digital tag” to individual transcripts. The 10-base digital tags are used to identify DNAs that are preferentially
amplified during PCR and count these species only once (i.e. as a single read count) during the analysis.

For analysis of the position of the RNAP trailing edge we identified reads from the 3’ ends of primer extension products that emanated from positions 4-8 bp upstream of the promoter -10 element. The mean position of the RNAP trailing edge was determined for each of the ~16,000 discriminator sequences using the formula below:

\[
\text{mean position of the RNAP trailing edge} = \frac{(-8 \times (\text{sum of the read count at position -8})) + (-7 \times (\text{sum of the read count at position -7})) + (-6 \times (\text{sum of the read count at position -6})) + (-5 \times (\text{sum of the read count at position -5})) + (-4 \times (\text{sum of the read count at position -4})))}{(\text{sum of the read count at positions -8 to -4})}
\]

For analysis of the position of the RNAP leading edge we identified reads from the 3’ ends of primer extension products that emanate from positions 12-16 bp downstream of the promoter -10 element. The mean position of the RNAP leading edge was determined for each of the ~16,000 sequences associated with each discriminator using the formula below:

\[
\text{mean position of the RNAP leading edge} = \frac{(12 \times (\text{sum of the read count at position 12})) + (13 \times (\text{sum of the read count at position 13})) + (14 \times (\text{sum of the read count at position 14})) + (15 \times (\text{sum of the read count at position 15})) + (16 \times (\text{sum of the read count at position 16})))}{(\text{sum of the read count at positions 12 to 16})}
\]

To establish whether any differences in the mean RNAP trailing-edge/leading-edge distance for distinct discriminator sequences were significant we compared the mean
RNAP trailing-edge/leading-edge distance for each of the 64 discriminator sequences to the mean RNAP trailing-edge/leading-edge distance value for each of the other 63 discriminator sequences by bootstrapping 10,000 subsets of the data for each discriminator sequence. The differences in the mean RNAP trailing-edge/leading-edge distance for each subset were fit to a Gaussian distribution using maximum likelihood estimation, and the fitness was evaluated using Kolmogorov–Smirnov test. Using the fitted distribution, the p-value of the two discriminator sequences being compared having a significant difference in the mean RNAP trailing-edge/leading-edge distance was calculated. The p-values of multiple comparisons were adjusted using Benjamini–Hochberg procedure.

3.4 Results

3.4.1 MASTER Analysis of TSS Selection Using Non-supercoiled Linear DNA

To analyze TSS selection, we considered only sequencing reads with 5' end sequences that precisely matched the sequence of the TSS region from which the transcript was generated, “matched RNAs”. For each TSS-region sequence, we calculated the number of matched RNAs emanating from each position 4–10 bp downstream of the -10 element (hereafter termed positions 4, 5, 6, 7, 8, 9, and 10). Next, we calculated the percentage of reads derived from matched RNAs emanating from each position (X) within the TSS region (% TSSX = 100 * [# reads at position X] / [# reads at all positions 4–10]) and used the %TSSX values to calculate a mean TSS for each sequence variant. The results define the full TSS inventories (“TSSomes”) from a consensus core promoter for non-
super-coiled templates in vitro, negatively supercoiled templates in vitro, and negatively supercoiled templates in vivo.

Averaging the %TSS at positions 4–10 observed for all TSS-sequence variants in reactions performed using non-supercoiled linear DNA templates shows, consistent with previous analyses of individual promoters (Aoyama and Takanami 1985; Sørensen et al. 1993; Jeong and Kang 1994; Liu and Turnbough 1994; Walker and Osuna 2002; Lewis and Adhya 2004), that TSS selection is determined by position (Figure 3.1A) (range = positions 6–10; mean = 7.36 bp downstream of 10 element; mode = 7 bp downstream of 10 element). The results further show the order of preference for TSS selection for non-supercoiled linear templates in vitro is position 7 > 8 > 6 > 9 > 10 (Figure 3.1A and Figure 3.1B; p < 0.001).
Figure 3.1. TSS Selection on a Non-supercoiled Linear DNA Template In Vitro. (A) TSS-distribution histogram. Average %TSS at positions 4–10 for TSS regions with R ≥ 25 matched RNA reads. (B) Sequence determinants for TSS selection. Table lists the amount of the total %TSS at positions 6–10 derived from TSS regions carrying (1) R or Y at the indicated TSS position; (2) A, G, C, or T at the indicated TSS position; or (3) Y_{TSS-1}R_{TSS} or R_{TSS-1}R_{TSS} at the indicated TSS position. (C) Sequence preferences for TSS selection. Sequence logo for the 162 TSS-region sequences (top 1%) with the highest %TSS at positions 6–10. Red bases indicate the TSS.
To investigate sequence determinants for TSS selection, we sorted TSS-region sequences on the basis of the identities of the bases at positions 6–10. As prior analyses of TSS selection have noted a bias for purine at the TSS position (Maitra and Hurwitz 1965; Jorgensen, Buch, and Nierlich 1969), we first determined the %TSS for promoter variants that specify use of a purine (R) TSS or pyrimidine (Y) TSS at positions 6–10. The results show a strong preference for R over Y at each TSS position (Figure 3.1B; p < 0.001). Further analysis of TSS region variants carrying an A, G, C, or T at each TSS position revealed the order of preference as G > A > C > T (Figure 3.1B; p < 0.001). Prior analyses of TSS selection also noted a preference for initiation at Y_{TSS-1}R_{TSS} sequence motifs (Shultzaberger et al. 2007). To investigate the influence of Y_{TSS-1}R_{TSS} sequences on TSS selection, we determined the %TSS for promoter variants carrying Y_{TSS-1}R_{TSS} or R_{TSS-1}R_{TSS}. The results show that Y_{TSS-1}R_{TSS} is preferred over R_{TSS-1}R_{TSS} at each TSS position (Figure 3.1B; p < 0.001).

We next identified TSS-region sequences that yielded the highest %TSS at each of the five positions within the TSS range (Figure 3.1C). For each TSS position, sequences that favor the highest %TSS exhibit a strong preference for Y_{TSS-1}R_{TSS}. In addition, for TSS position 9 there is a strong preference for T_{TSS-1}, while for TSS position 10, the sequences with the highest %TSS show a preference for a T_{TSS-2}T_{TSS-1}.

3.4.2 MASTER Analysis of TSS Selection Using Negatively Supercoiled DNA

For analysis of the effects of DNA topology on TSS selection, we compared results obtained in vitro using either a non-supercoiled linear DNA template or a relaxed circular DNA template with results obtained using a negatively supercoiled DNA template in vitro.
and in vivo. Analysis of the %TSS at positions 4–10 observed for all TSS-sequence variants in reactions performed using negatively supercoiled DNA templates (Figure 3.2A and Figure 3.2C) shows that the range of TSS positions (positions 6–10) and modal TSS (position 7) was identical to the range of TSS positions and modal TSS observed using a non-supercoiled DNA template (Figure 3.1A).

In addition, the sequence determinants for TSS selection observed using negatively supercoiled DNA templates were identical to those observed using non-supercoiled DNA templates (Figure 3.1B and Figure 3.1C). In particular, R is favored over Y at each TSS position, the order of TSS preference for each base is G > A > C ~ T, and Y_{TSS-1}R_{TSS} is preferred over R_{TSS-1}R_{TSS} at each TSS position (p < 0.001). Furthermore, the TSS-region sequences that yielded the highest %TSS at each of the five TSS positions within the TSS range were similar on negatively supercoiled DNA templates and non-supercoiled DNA templates (Figure 3.1C).

However, there were also notable differences between the results obtained using negatively supercoiled DNA templates and non-supercoiled DNA templates. First, while the %TSS observed at position 6 and position 9 were similar with non-supercoiled DNA (Figure 3.1A and Figure 3.1B), the %TSS at position 9 was ~3 times larger than the %TSS at position 6 with negatively supercoiled DNA (Figure 3.2A and Figure 3.2C). Second, comparison of the mean TSS observed with negatively supercoiled DNA (Figure 3.2A and Figure 3.2C) with the mean TSS observed using non-supercoiled DNA (Figure 3.2A) reveals TSS distributions with negatively supercoiled DNA (mean = 7.65 bp downstream of 10 element with negatively supercoiled DNA in vitro; mean = 7.59 bp downstream of 10 element with negatively supercoiled DNA in vivo) are shifted downstream relative to TSS distribution with non-supercoiled DNA (mean = 7.36 bp
downstream of 10 element). In addition, 95% of TSS-region sequence variants exhibited an increase in the mean TSS with negatively supercoiled DNA templates compared with a non-supercoiled DNA template (Figure 3.2B and Figure 3.2D).

Analysis of the effects of TSS-region sequence on sensitivity to topology revealed that TSS-region sequences carrying R at positions 7 and 8 are less susceptible to topology effects, while TSS-region sequences carrying a Y at positions 7 and 8 are more susceptible to topology effects (Figure 3.2E; p < 0.001). In addition, TSS-region sequences that contained YR at positions 6/7 are less susceptible to topology effects, while TSS-region sequences carrying a RY at positions 6/7 are more susceptible to topology effects (Figure 3.2E; p < 0.001). Furthermore, identification of the TSS-region sequences that exhibited the highest difference in mean TSS in a comparison of negatively supercoiled DNA and non-supercoiled DNA revealed that TSS-region sequences that contained RYYR at positions 6–9 or RYYYYR at positions 6–10 were highly susceptible to topology-dependent changes in TSS selection (Figure 3.2E and Figure 3.2F; p < 0.001).

The results obtained from our MASTER analysis of TSS selection on non-supercoiled and negatively supercoiled DNA indicate that DNA topology is a determinant of TSS selection and reveal TSS-region sequence determinants that confer sensitivity or resistance to topology-dependent changes in TSS selection for a consensus core promoter. In addition, a comparison of the results obtained using negatively supercoiled DNA in vitro (Figure 3.2A) with results obtained using negatively supercoiled DNA in vivo (Figure 3.2C) show that TSS selection with negatively supercoiled DNA in vitro accurately recapitulates TSS selection with negatively supercoiled DNA in vivo, suggesting that all determinants for TSS selection for a consensus core promoter are
contained within an in vitro transcription reaction (i.e., RNAP, DNA, and NTPs) and that no other determinants or factors have major effects on global TSS distributions in vivo (Figure 3.2C) or on global sequence determinants for TSS selection in vivo.

Figure 3.2. TSS Selection on Negatively Supercoiled DNA Templates. (A) TSS-distribution histogram for experiments performed in vitro. Average %TSS at positions 4–10 for TSS regions with ≥ 25 matched RNA reads. (B) Plot of the mean TSS with...
negatively supercoiled DNA in vitro versus the mean TSS with non-supercoiled linear DNA in vitro for individual TSS-region sequences. (C) TSS-distribution histogram for experiments performed in vivo. Average %TSS at positions 4–10 for TSS regions with ≥ 25 matched RNA reads. (D) Plot of the mean TSS with negatively supercoiled DNA in vivo versus the mean TSS with non-supercoiled linear DNA in vitro for individual TSS-region sequences. (E) Average of the mean TSS values for the indicated TSS-region sequences (Δ mean TSS; differences between values observed on linear and supercoiled templates). (F) Sequence preferences for topology-dependent effects on TSS selection. Sequence logo and average mean TSS values for 162 TSS-region sequences (top 1%) with the highest values of Δ mean TSS.

3.4.3 MASTER Analysis of Effects of NTP Concentrations on TSS Selection

To analyze the effects of NTP concentrations on TSS selection, we compared the results obtained from the analysis of in vitro reactions performed in the presence of saturating NTPs (2.5 mM NTPs:Mg$^{2+}$) with results obtained from the analysis of in vitro reactions performed in the presence of non-saturating NTPs (0.1 mM). Results show that the range of TSS positions (6–10) and modal TSS (position 7) are identical at saturating and non-saturating NTP concentrations (Figure 3.3A and Figure 3.3B) and identical to the range and mode observed in the analysis of non-supercoiled and supercoiled DNA templates in vitro at 1 mM NTPs and the analysis of supercoiled DNA templates in vivo (Figure 3.1A, Figure 3.2A, and Figure 3.2C). Comparison of the sequence determinants for TSS selection at saturating and non-saturating NTP concentrations revealed that preference for an R TSS over a Y TSS increases at non-saturating NTP
concentrations, preference for a G TSS increases at non-saturating NTP concentrations, and preference for \( Y_{\text{TSS}} R_{\text{TSS}} \) over \( R_{\text{TSS}} R_{\text{TSS}} \) increases at non-saturating NTP concentrations (Figure 3.3C and Figure 3.3D).

Comparison of the mean TSS observed at saturating and non-saturating NTP concentrations (Figure 3.3A and Figure 3.3B) reveals the overall TSS distribution at non-saturating NTP concentrations (mean = 7.50 bp downstream of −10 element) is shifted slightly downstream relative to the overall TSS distribution at saturating NTP concentrations (mean = 7.38 bp downstream of −10 element). Comparison of individual TSS-sequence variants revealed that a majority of TSS-sequence variants exhibited a change in the mean TSS at non-saturating NTP concentrations compared with saturating NTP concentrations (Figure 3.3E). TSS-region sequences carrying an R at position 7 and position 8 are less susceptible to alterations in TSS selection in response to alterations in NTP concentrations, while TSS-region sequences carrying a Y at position 7 and position 8 are more susceptible to alterations in TSS selection in response to alterations in NTP concentrations (Figure 3.3F; \( p < 0.001 \)). In addition, TSS-region sequences that contained YR at positions 6/7 are less susceptible to the effects of alterations in NTP concentrations, while TSS-region sequences carrying a RY at positions 6/7 are more susceptible to the effects of alterations in NTP concentrations (Figure 3.3F; \( p < 0.001 \)). Furthermore, TSS-region sequences that contained RYYR at positions 6–9 or RYYYR at positions 6–10 were highly susceptible to NTP-concentration dependent changes in TSS selection (Figure 3.3F; \( p < 0.001 \)). Strikingly, the patterns observed in the comparison of sequences that are more or less susceptible to NTP-concentration dependent changes in TSS selection (Figure 3.3F) were similar to the patterns observed in the comparison of sequences that are more or less susceptible to topology-dependent changes in TSS selection (Figure 3.2E).
Figure 3.3. Effects of NTP Concentrations on TSS Selection In Vitro. (A and B) TSS-distribution histograms at saturating (A) and non-saturating (B) NTP concentrations in vitro. Average %TSS at positions 4–10 for TSS regions with ≥ 25 matched RNA reads. Experiments were performed at 2.5 mM NTPs:Mg2+ (saturating) or 0.1 mM NTPs (non-saturating) using a non-supercoiled linear DNA template. (C and D) Sequence...
determinants for TSS selection (C, saturating NTPs; D, non-saturating NTPs). (E) Plot of the mean TSS at saturating NTP concentrations versus non-saturating NTP concentrations for individual TSS-region sequences. (F) Average of the mean TSS values observed for the indicated TSS-region sequences at saturating (sat.) and non-saturating (non-sat.) NTP concentrations (Δ mean TSS; differences between values observed at saturating and non-saturating NTP concentrations).

3.4.4 MASTER Analysis Defines the Extent of Productive Slippage Synthesis during Transcription Initiation

During the standard pathway of transcription initiation, in each nucleotide addition step, RNAP translocates relative to the DNA and RNA, and the DNA template strand and the 3’ end of the RNA product remain in register (Figure 3.4A). However, during initial transcription RNAP can enter into an alternative pathway of transcription termed “slippage synthesis” (Jacques and Kolakofsky 1991; Turnbough and Switzer 2008). In slippage synthesis, RNAP does not translocate relative to the DNA and RNA, and instead the RNA product slips upstream relative to the DNA template strand, establishing a new register of DNA and RNA (Figure 3.4B). Slippage synthesis can occur in multiple cycles, including very large numbers of cycles. Accordingly, slippage synthesis is also referred to as “reiterative transcription initiation”.

RNAs produced as a consequence of slippage synthesis either can be released from the initial transcribing complex (“non-productive slippage synthesis”) or can be extended to yield full-length RNA products (“productive slippage synthesis”). Full-length RNA products generated by productive slippage synthesis typically contain at least one 5’ nucleotide that does not match the sequence of the DNA template (“RNA/DNA
Therefore, to identify products of productive slippage synthesis in our analysis of transcription output, we enumerated reads that carried at least one RNA/DNA difference, and since slippage synthesis can occur in multiple cycles, we also enumerated reads with 5' ends that were up to five bases longer than position 4, i.e., the first position of the randomized TSS region.

To assess the extent of productive slippage synthesis, we analyzed the total transcription output from each TSS-region sequence variant for transcripts predicted to be generated by productive slippage synthesis.

First, we assessed the extent of productive slippage synthesis at promoters that contain homopolymeric repeat sequences that start at the TSS (i.e., T_n, A_n, G_n, and C_n, where n > 1 and where the sequence starts at the TSS). Such sequences potentially are expected to facilitate productive slippage synthesis, in increments of one base pair, since homopolymeric repeat sequences allow slippage to occur, in increments of one base pair, with a net loss of only one RNA-DNA base pair (Figure 3.4B). Products of standard synthesis and productive slippage synthesis from homopolymeric repeat sequences were distinguished by the absence or presence, respectively, of at least one RNA/DNA difference. For example, for the TSS-region sequence AAT, which carries an A_2 homopolymeric repeat starting at the underlined TSS base, RNAs generated by standard synthesis would have the fully templated 5' end sequence AAU-, whereas potential RNAs generated by productive slippage synthesis would have 5' end sequences AAAU-, AAAAAU-, AAAAAAU-, etc. that carry one or more 5' RNA/DNA difference. We calculated the percentage of RNAs from the homopolymeric tract that are produced by slippage (% slippage = 100[#slippage reads] / [#slippage reads + #standard reads]). The results indicate that: (1) slippage occurs at promoters that contain T_n, A_n,
Cₙ, and Gₙ homopolymeric repeats at TSS positions 6, 7, 8, and 9 (Figure 3.4C); (2) % slippage is especially high for Tₙ and Aₙ homopolymeric repeats (up to > 80%; Figure 3.4C); (3) % slippage increases in all cases as the length of a Tₙ, Aₙ, and Gₙ homopolymeric repeat increases and increases in many cases as the length of a Cₙ homopolymeric repeat increases (Figure 3.4C); (4) the number of nucleotides added to the RNA 5' end by repeated cycles of slippage increases as the length of a Tₙ, Aₙ, and Gₙ homopolymeric repeat increases; and (5) the number of nucleotides added to the 5' end by repeated cycles of slippage can be strikingly long (up to at least eight for Tₙ and Aₙ homopolymeric repeats).
Figure 3.4. Comprehensive Analysis of Productive Slippage Synthesis. (A) Nucleotide addition cycle for the standard pathway of transcription initiation. Left: initial transcribing complex with a 2-nt RNA in a pre-translocated state. Middle: initial transcribing complex with a 2-nt RNA in a post-translocated state. Right: 3-nt product complex in a pre-
translocated state. The RNA and DNA template strand remain in lock-step register, and the sequence of the RNA is fully complementary to the template strand. White boxes, DNA; blue boxes, RNA; gray shading, RNAP; red, TSS bases; i and i+1, RNAP active-center i and i+1 sites. (B) Nucleotide addition cycle for the slippage pathway. Left: initial transcribing complex with a 2-nt RNA in a pre-translocated state. Middle: RNA has moved backward relative to the DNA template by one base. Right: 3-nt product complex in a pre-translocated state. The 5’ end of the RNA carries an RNA/DNA difference and is not complementary to the template strand. (C) Analysis of productive slippage synthesis. Graphs show % slippage (mean + SEM) for TSS-region sequences containing 5’ end homopolymeric repeat sequences of the indicated length that begin at the indicated position (TSS).

3.4.5 MASTER Analysis Reveals Effects of TSS-Region Sequences on Transcript Yield

To document effects of TSS-region sequence on relative transcript yield, we divided the read count representing the total transcription output for each TSS-region sequence variant by the relative number of DNA templates that carried the sequence variant. We refer to the value obtained as the “relative expression” of each TSS-region sequence variant. A comparison of the relative expression values allowed us to determine the influence of sequence variation in the TSS region on the range of expression observed for a given experimental condition (Figure 3.5A and Figure 3.5B).

For experiments performed in vitro using non-supercoiled linear DNA or negatively supercoiled DNA (at 1 mM NTPs) TSS-region sequence variation led to a ~40-fold range
of relative expression. For experiments performed with negatively supercoiled DNA in vivo, we found that TSS-region sequence variation led to a >100-fold range of relative expression. However, interpreting the effects of TSS-region sequences on transcript yields in vivo is complicated by the potential contribution of sequence variation at the RNA 5’ end to the stability of full-length transcripts. Thus, we infer that the analysis of the effects of TSS-region sequence variation on relative transcript yield in vitro provides a more accurate measure of the true impact of TSS-region sequence variation on transcription output. We therefore conclude that DNA topology does not exert a significant global impact on the relationship between TSS-region sequence and the range of expression.

We next compared the effect of NTP concentrations on the range of expression. For assays performed in vitro at saturating NTP concentrations, TSS-region sequence variation led to a ~14-fold range of relative expression (Figure 3.5A). In contrast, for assays performed in vitro at non-saturating NTP concentrations, TSS-region sequence variation led to a ~100-fold range of relative expression (Figure 3.5B). Thus, the results show that limiting the concentrations of NTPs significantly enhances effects of TSS-region sequence variation on expression. Results further show that the magnitude of the difference in relative expression of TSS-region sequences carrying an R at TSS positions 7 and 8 relative to TSS-region sequence variants carrying a Y at TSS positions 7 and 8 increased at non-saturating NTP concentrations (Figure 3.5C). In addition, the magnitude of the difference in relative expression observed from TSS-region sequences carrying YR versus RY at positions 6/7 also increased at non-saturating NTP concentrations (Figure 3.5C). Furthermore, TSS-region sequences carrying 2–4 consecutive T bases, which exhibit high % slippage (Figure 3.4), exhibit a large decrease in relative expression at non-saturating NTP concentrations compared with
saturating NTP concentrations (Figure 3.5C). Thus, we propose that this decrease in relative expression occurs, at least in part, due to a decrease in productive slippage synthesis and concomitant increase in non-productive slippage synthesis (undetectable by MASTER), as a consequence of performing reactions at non-saturating NTP concentrations.

Our analysis of the effects of TSS-region sequence on transcript yield shows that sequence variation in the TSS region can impact overall transcript yields (i.e., promoter strength) by at least two orders of magnitude. We conclude that the TSS-region sequence is a key determinant of promoter strength.
Figure 3.5. Effects of NTP Concentrations on Transcript Yields In Vitro. (A and B)
Relative expression histograms for experiments performed at saturating NTP (A) and
non-saturating (B) NTP concentrations using a non-supercoiled linear DNA template in
vitro. Relative expression for TSS-region sequences with ≥ 25 total RNA reads for which
the number of DNA templates was not in the top or bottom 10%. For each experimental
condition, the lowest value of relative expression was normalized to 1. (C) Normalized
relative expression for the indicated TSS-region sequences. Values were calculated by
dividing the average relative expression for the indicated TSS-region sequence by the
relative expression observed for all TSS-region sequences.

3.4.6 5’ mNET-seq analysis of Sequence-Specific RNAP–CRE

Interactions in E. coli Transcriptome In Vivo

We used mNET-seq to define the contribution of sequence-specific RNAP–G<sub>CRE</sub>
interactions to TSS selection in vivo (Vvedenskaya et al. 2014). G<sub>CRE</sub> is the non-
template-strand guanine nucleotide at the extreme downstream end of the transcription
bubble, and the position of G<sub>CRE</sub> is denoted as TSS+1<sub>NT</sub> in the following text. mNET-seq
involves selective analysis of transcripts associated with an epitope-tagged RNAP in the
presence of a mixed population of epitope-tagged RNAP and untagged RNAP. We used
5’ mNET-seq to determine the effect of sequence-specific RNAP–G<sub>CRE</sub> interactions on
TSS selection. To do this, we introduced into cells a plasmid encoding 3xFLAG-tagged
β<sup>WT</sup> or 3xFLAG-tagged β<sup>D446A</sup>, isolated RNA products associated with RNAP-β<sup>WT</sup> or
RNAP-β<sup>D446A</sup> by immunoprecipitation, converted RNA 5’ ends to cDNAs, and performed
high-throughput sequencing.
Figure 3.6. Effects of disrupting RNAP–GCRE interactions in vivo. (A and B) 5’ mNET-seq analysis of 47 (~16,000) consensus promoter derivatives. (A) Effect of sequence at TSS+1_{NT} on %TSS for RNAP-β^{WT} vs. RNAP-β^{D446A}. Table lists the difference in %TSS (%TSS for RNAP-β^{WT} − %TSS for RNAP-β^{D446A}) at positions 6, 7, 8, or 9 for TSS-regions carrying G, A, C, or T at TSS+1_{NT}. (B) Sequence preferences for TSS+1_{NT}. Sequence logo for TSS+1_{NT} of above-threshold TSS positions located 6–9 bp downstream of the −10 element (Left) and TSS positions located 6–9 bp downstream of the −10 element that exhibited a large, ≥20%, reduction in %TSS in 5’ mNET-seq analysis of RNAP-β^{D446A} vs. 5’ mNET-seq analysis of RNAP-β^{WT} (Right). (C) 5’ mNET-seq analysis of E. coli transcriptome. Sequence preferences for TSS+1_{NT}. Sequence logo for TSS+1_{NT} of above-threshold TSS associated with natural promoters (Left) and TSS associated with
natural promoters that exhibited a large, ≥20%, reduction in %TSS in 5’ mNET-seq analysis of RNAP-β^{D446A} vs. RNAP-β^{WT}.

To enable direct comparison of in vivo and in vitro results, we performed 5’ mNET-seq using the same MASTER template library of $4^7$ (~16,000) consensus core promoter derivatives that we used for in vitro analysis. For all four tested TSS positions (positions 6, 7, 8, and 9), the β^{D446A} substitution decreased the %TSS when TSS+1NT was G (0.6–7.3% decreases) (Figure 3.6A, top row of table). In contrast, for three of the four tested TSS positions (positions 6, 7, and 8), the β^{D446A} substitution did not decrease the %TSS when TSS+1_{NT} was A, C, or T, and, for the fourth position (position 9), the β^{D446A} substitution decreased the %TSS by smaller amounts when TSS+1_{NT} was A, C, or T (Figure 3.6A, bottom three rows of table). Furthermore, we identified 860 TSS positions (4.3% of the 20,217 above-threshold TSS positions located 6-, 7-, 8-, or 9-bp downstream of the −10 element) with large, ≥20%, CRE effects. For these 860 TSS positions with large, ≥20%, CRE effects, ~80% contained G at TSS+1_{NT} (Figure 3.6B, Right), whereas, for the total sample of 20,217 TSS positions, there were no detectable sequence preferences at position TSS+1_{NT} (Figure 3.6B, Left).

Having shown by MASTER promoter sequence variants that sequence-specific RNAP–CRE interactions are a determinant of TSS selection in the context of a consensus core promoter in vivo, we next assessed the contribution of sequence-specific RNAP–CRE interactions to TSS selection in the context of natural promoters in vivo in E. coli. Using data from experiments performed with RNAP-β^{WT}, we identified 1,500 above-threshold TSS positions associated with natural promoters in E. coli. Of these 1,500 TSS positions, we identified 44 TSS positions that exhibited large, ≥20%, CRE effects; 39 of
these 44 (∼90%) contained G at TSS+1NT (Figure 3.6C, Right), whereas for the total sample of 1,500 above-threshold TSS, there were no detectable sequence preferences at TSS+1NT (Figure 3.6C, Left).

The results establish that disrupting sequence-specific RNAP–G_CRE interactions affects TSS selection in vivo in a manner that correlates with the presence and position of G_CRE in the TSS region. We conclude that sequence-specific RNAP–CRE interactions are a determinant of TSS selection in vivo.

3.4.7 CapZyme-Seq Analysis of NCIN Capping with NAD⁺ In Vitro

To define promoter-sequence determinants for NCIN capping, we combined CapZyme-seq with MASTER (Figure 3.7A). MASTER enables measurement of RNA 5' end sequences and RNA yields for RNAs generated during transcription of a template library of up to at least 4¹⁰ barcoded template sequences (Vvedenskaya et al. 2015; Winkelman et al. 2016). Accordingly, combining CapZyme-seq with MASTER enables measurement of RNA 5' end sequences and RNA yields for both NCIN-capped RNA and uncapped, 5'-triphosphate RNA for each of up to at least 4¹⁰ promoter sequences (Figure 3.7A).
Figure 3.7. CapZyme-Seq Analysis of NCIN Capping with NAD⁺ In Vitro. (A) Use of CapZyme-seq in combination with massively systematic transcript end readout (MASTER). Top, lacCONS-N7 promoter library (4^7, ~16,000 promoter sequences). Gray, promoter −35 and −10 elements; green, randomized sequences 4–10 bp downstream of promoter −10 element; blue, transcribed-region barcode. The linear DNA template contains ~100 bp of transcribed-region sequence downstream of the green randomized sequence. Thus, RNA products generated from the lacCONS-N7 promoter library are ~100 nt in length. Middle, CapZyme-seq using NudC for processing of NCIN-capped RNA and Rpp for processing of uncapped 5’-triphosphate RNA (CapZyme-seq^NudC). Bottom, CapZyme-seq using Rai1 for processing of NCIN-capped RNA and
Rpp for processing of uncapped 5′-triphosphate RNA (CapZyme-seq\textsuperscript{Rai1}). (B) Equations used to calculate percent capping and capping efficiencies. (C) Results of CapZyme-seq\textsuperscript{NudC} and CapZyme-seq\textsuperscript{Rai1}. Top: mean percent capping from CapZyme-seq\textsuperscript{NudC} (n = 3) versus mean percent capping from CapZyme-seq\textsuperscript{Rai1} (n = 3) for ~16,000 promoter sequences (density from Gaussian kernel density estimation method). Middle and bottom: percent capping histograms.

We used the MASTER promoter library \textit{lacCONS}-N7 (Vvedenskaya et al. 2015), which contains $4^{7}$ (∼16,000) derivatives of a consensus \textit{E. coli} σ70-dependent promoter comprising all possible sequence variants at the positions 4 to 10 base pairs (bp) downstream of the promoter −10 element (positions 4, 5, 6, 7, 8, 9 and 10; \textbf{Figure 3.7A}, top). We performed in vitro transcription experiments using the \textit{lacCONS}-N7 promoter library and \textit{E. coli} RNAP σ70 holoenzyme, in parallel, in the presence or absence of NAD\textsuperscript{+}. RNA products from each reaction were analyzed with CapZyme-seq using NudC (CapZyme-seq\textsuperscript{NudC}; \textbf{Figure 3.7A}, middle) or Rai1 (CapZyme-seq\textsuperscript{Rai1}; \textbf{Figure 3.7A}, bottom). We determined percent capping (capped RNA yields relative to total RNA yields) and capping efficiency (NAD\textsuperscript{+}-mediated initiation relative to NTP-mediated initiation; $(K_{cat}/K_{m, \text{NAD}})/(K_{cat}/K_{m, \text{NTP}})$) from the resulting RNA yields using the equations in \textbf{Figure 3.7B}. Comparison of results obtained using CapZyme-seq\textsuperscript{NudC} with results obtained using CapZyme-seq\textsuperscript{Rai1} indicates the results are well correlated ($r^2$ ∼0.95; slope ∼1.0; \textbf{Figure 3.7C}, top). The mean percent capping observed is ∼23%, the median percent capping is ∼10%, and the range of percent capping is 0%–95% for the $4^{7}$ promoter sequences (\textbf{Figure 3.7C}, bottom). The majority of percent capping values fall within the range of 0%–15%. The distribution of percent capping values is highly skewed with a high peak of 0%–5% and a long tail extending to greater than 90%.
The skewed, long-tailed distribution of percent capping confirms that different promoter sequences differ markedly in efficiency of NCIN capping with NAD$^+$.}

3.4.8 Determinants for TSS Selection in NCIN Capping with NAD$^+$ In Vitro

In bacterial transcription initiation, RNAP selects a TSS at a variable distance downstream of the promoter −10 element. According to the results from MASTER analysis of TSS selection in NTP-mediated initiation with the lacCONS-N7 promoter library (Figure 2), TSS selection occurs over a range of five positions located 6–10 bp downstream of the −10 element (positions 6, 7, 8, 9, and 10); the preferred, modal position for TSS selection is position 7; and the order of preference for TSS selection is 7 > 8 > 9 > 6 > 10. Results further indicated that there is a strong sequence preference for G or A (R) at each TSS position.

To define, comprehensively, the determinants for TSS selection in NAD$^+$-mediated initiation, we used the combination of CapZyme-seq and MASTER to determine the 5' end sequence and yields of NAD$^+$-capped RNA in NAD+-mediated initiation with the lacCONS-N7 promoter library (Figure 3.8A and Figure 3.8B). To compare these determinants with determinants for TSS selection in NTP-mediated initiation under identical reaction conditions, we used the combination of CapZyme-seq and MASTER to determine the 5' end sequence and yields of uncapped, 5'-triphosphate RNA in NTP-mediated initiation with the lacCONS-N7 promoter library in the presence or absence of NAD$^+$ (Figure 3.8C). As with the results above for percent capping (Figure 3.7C), the
results here for TSS selection obtained using CapZyme-seq$^{NudC}$ and CapZyme-seq$^{Rai1}$ are well correlated ($r^2 \sim 0.95$; slope $\sim 1.0$; **Figure 3.8B and Figure 3.8C**). The positional preferences for TSS selection in NAD$^+$-mediated initiation (range = positions 6–10; mode = 7 bp downstream of −10 element; mean 7.5 bp downstream of −10 element; order of preference = 7 > 8 > 9 > 6 > 10; **Figure 3.8B**, middle) are indistinguishable from the positional preferences for TSS selection in NTP-mediated initiation (range = positions 6–10; mode = 7 bp downstream of −10 element; mean 7.6 bp downstream of −10 element; order of preference = 7 > 8 > 9 > 6 > 10 in reactions performed both in the presence or absence of NAD$^+$; **Figure 3.8C**, middle). However, the sequence preferences for TSS selection in NAD$^+$-mediated initiation differ from the sequence preferences for TSS selection in NTP-mediated initiation, exhibiting an essentially absolute preference for TSS positions where the base pair is A:T (**Figure 3.8B**, bottom) instead of a preference for TSS positions where the base pair is either A:T or G:C (**Figure 3.8C**, bottom), consistent with expectations based on the base pairing preferences of the adenosine moiety of NAD$^+$. 
Figure 3.8. Determinants for TSS Selection in NCIN Capping with NAD+ In Vitro. (A) lacCONS-N7 promoter library (4^7, ~16,000 promoter sequences). (B and C) Data for NAD⁺-mediated initiation (B) and NTP-mediated initiation (C). Top: mean TSS from CapZyme-seq^NudC (n = 3) versus mean TSS from CapZyme-seq^Rai1 (n = 3; mean TSS = [(4 x %TSS at position 4) + (5 x %TSS at position 5) + (6 x %TSS at position 6) + (7 x %TSS at position 7) + (8 x %TSS at position 8) + (9 x %TSS at position 9) + (10 x %TSS at position 10)]/3) versus mean TSS from CapZyme-seq^Rai1. Bottom: %TSS distribution at each position (mean ± SD).
3.4.9 Promoter Sequence Determinants for NCIN Capping with NAD$^+$

In Vitro

The results in the previous section show that the modal, consensus TSS position for NAD$^+$-mediated initiation is 7 bp downstream of the promoter −10 element and that the consensus TSS base pair for NAD$^+$-mediated initiation is A:T. Considering the subset of ~4,000 promoter sequences in the lacCONS-N7 promoter library that have A:T at the modal, consensus TSS position for NAD$^+$-mediated initiation, 7 bp downstream of the promoter −10 element (A$^{+1}$ promoters), we next assessed promoter sequence determinants for NAD$^+$-mediated initiation at each of the three positions upstream of the TSS (positions 4, 5, and 6 bp downstream of the −10 element; positions −3, −2, and −1 relative to the TSS, position +1; Figure 3.9) and at each of the three positions downstream of the TSS (positions 8, 9, and 10 bp downstream of the −10 element; positions +2, +3, and +4 relative to the TSS; Figure 3.9). We find that capping efficiency depends on the identity of the nucleotide at each of these positions (Figure 3.9B and Figure 3.9C). At position −3, capping efficiency is higher for A, T, and C than for G, yielding the consensus H and anti-consensus G; at position −2, capping efficiency is higher for G and A than for T and C, yielding the consensus R and anti-consensus Y; at position −1, capping efficiency is higher for G and A than for T and C, yielding the consensus R; at position +2, capping efficiency is higher for G and C than for A and T, yielding the consensus S; at position +3, capping efficiency is higher for A and T than for
G and C, yielding the consensus W and anti-consensus S; and at position +4, capping efficiency is higher for A and T than for G and C, yielding the consensus W and anti-consensus S (Figure 3.9C). The strongest dependence of capping efficiency on nucleotide identity, at positions other than the TSS, is observed at position −1. At this position, the mean relative capping efficiencies for promoters having A and G are about 2 to 3 times higher than for promoters having T and about 7 to 8 times higher than for promoters having C (Figure 3.9C). The second-strongest dependence of capping efficiency on nucleotide identity is observed at position +2. At this position, capping efficiencies for promoters having G and C are ~2 to ~3 times higher than for promoters having A and T (Figure 3.9C). The dependencies of capping efficiencies on nucleotide identity at each of the other positions (−3, −2, +2, +3, and +4) are smaller but significant (Figure 3.9C).

The finding that position −1 is a crucial sequence determinant, with G or A as the preferred nucleotides, confirms our previous results (Bird et al. 2016) and contradicts the results of Julius and Yuzenkova (Julius and Yuzenkova 2017). Julius and Yuzenkova (Julius and Yuzenkova 2017) did not observe specificity at position −1, most likely because they measured only 1/K_m and not k_{cat}/K_m and thus were unable to detect specificity manifest at the level of k_{cat}. The findings for sequence specificity at positions −3, −2, +2, +3, and +4 are new to this work.
Figure 3.9. Promoter Sequence Determinants for NCIN Capping with NAD⁺ In Vitro. (A) Subset of lacCONS-N7 promoter library having A (red) at the position 7 bp downstream of −10 element (~4,000 sequences). (B) Distributions of relative capping efficiency (n = 6) for ~4,000 A⁺1 promoter sequences at the positions immediately upstream of the TSS (positions −1, −2, and −3) and immediately downstream of the TSS (positions +2, +3, and +4). The dashed line is the mean relative capping efficiency, the upper and lower solid lines are the 95th percentile and 5th percentile, respectively, and the range is defined as the 95th percentile relative capping efficiency divided by the 5th percentile relative capping efficiency. (C) Distributions of relative capping efficiency for ~4,000 A⁺1 promoter sequences parsed by position and nucleotide (A, T, C, or G). The dashed line is the mean relative capping efficiency for all sequences, the solid lines are the means.
for sequences having the indicated nucleotide. Distributions and lines are colored by consensus nucleotide (mean relative capping efficiency greater than 1; red) or anti-consensus nucleotide (mean relative capping efficiency less than 1; blue). Shown are the p values for pairwise comparisons of consensus and anti-consensus nucleotides (Kolmogorov-Smirnov test).

### 3.4.10 Promoter Consensus Sequence for NCIN Capping with NAD$^+$ In Vitro

The results in the previous section provide a promoter consensus and anti-consensus sequence for NCIN capping with NAD$^+$ of H$_{-3}$R$_{-2}$R$_{-1}$A$_{+1}$S$_{+2}$W$_{+3}$W$_{+4}$ and G$_{-3}$Y$_{-2}$Y$_{-1}$A$_{+1}$W$_{+2}$S$_{+3}$S$_{+4}$, respectively (Figure 3.10A). The results in Figure 3.10B indicate that the mean relative capping efficiencies for promoters having a consensus nucleotide at positions −3, −2, −1, +2, +3, and +4 are ~1.4-fold, ~1.5-fold, ~4.1-fold, ~2.6-fold, ~1.3-fold, and ~1.2-fold, respectively, greater than the mean relative capping efficiencies for promoters having an anti-consensus nucleotide at these positions. Mean capping efficiency values for consensus A$_{+1}$ promoter sequences versus anti-consensus A$_{+1}$ promoter sequences differ by ~40-fold (Figure 3.10C).
Figure 3.10. Promoter Consensus Sequence for NCIN Capping with NAD⁺ In Vitro. (A) lacCONS promoter derivatives with consensus A⁺₁ sequence (red) and anti-consensus A⁺₁ sequence (blue) for NAD⁺ capping. (B) Distributions of relative capping efficiency, for ~4,000 A⁺₁ promoter sequences parsed by position and nucleotide (H, G, R, Y, S, or W) and colored by consensus nucleotide (red) or anti-consensus nucleotide (blue). The dashed line is the mean relative capping efficiency for all sequences, the solid lines are the means for sequences having a consensus nucleotide (red) or an anti-consensus nucleotide (blue). (C) Distributions of relative capping efficiency for consensus A⁺₁ sequences (red), anti-consensus A⁺₁ sequences (blue), or all A⁺₁ sequences (gray).
3.4.11 Promoter-Sequence Determinants for NCIN Capping In Vivo

To determine total levels of NCIN capping and to define promoter-sequence determinants for NCIN capping we isolated RNA products from *E. coli* cells containing the MASTER lacCONS-N7 template library used in the experiments in Figure 3.7, Figure 3.8, Figure 3.9, and Figure 3.10 and analyzed RNA products from the 47 MASTER lacCONS-N7 promoter sequences using CapZyme-seqNudC (Figure 3.11A).

We observed NCIN capping with many promoter sequences in vivo, extending and generalizing our conclusion from previous work with a single promoter sequence (Bird et al. 2016). The level of NCIN capping differs for RNA products from the 47 different promoter sequences, ranging from 0 to 38%, with a mean of 3%, and a median of 2% (Figure 3.11A). We see a broad distribution of percent capping values in vivo (Figure 11A) reminiscent of the broad distribution of percent capping values observed in vitro (Figure 3.7C).

The preferred TSS positions for NCIN capping in vivo matches the preferred TSS positions for NCIN capping with NAD$^+$ in vitro (mode = 7 bp downstream of −10 element; mean = 7.5 bp downstream of −10 element; Figure 3.8B and Figure 3.11B). The preferred TSS base pair for NCIN capping in vivo (A:T) also matches the preferred TSS base pair for NCIN capping with NAD$^+$ in vitro (Figure 3.8B).

Considering the subset of ~4,000 promoter sequences in the lacCONS-N7 promoter library that have A:T at the modal, consensus TSS position for NCIN capping, 7 bp downstream of the promoter −10 element (A+1 promoters), we next assessed promoter sequence determinants for NCIN capping in vivo at each of the three positions upstream
and downstream of the TSS (Figure 3.11C). We find that for positions −3 to +2, the sequence determinants for NCIN capping in vivo (Figure 3.11C) match those for NCIN capping with NAD$^+$ in vitro (Figure 3.9 and Figure 3.10). At position +3, we observe no sequence preferences in vivo. At position +4, we observe sequence preferences similar to but weaker than those observed in vitro (Figure 3.9). The results provide promoter consensus and anti-consensus sequences for NCIN capping in vivo—H$_3$R$_2$R$_1$A$_1$S$_2$ and G$_3$Y$_2$Y$_1$A$_1$W$_2$ (Figure 3.11C and Figure 3.11D)—that match the corresponding promoter consensus and anti-consensus sequences for positions −3 to +2 for NCIN capping with NAD$^+$ in vitro (Figure 3.10). The strongest promoter sequence dependence of NCIN capping in vivo, apart from that at the TSS, is observed at position −1 (Figure 3.11C), matching the pattern observed for NCIN capping with NAD$^+$ in vitro (Figure 3.9B and Figure 3.9C). At this position, the difference, $\Delta$, in mean percent capping for consensus versus anti-consensus is $\sim$6.5% (Figure 3.11C). At positions −3, −2, and +2, the difference, $\Delta$, in mean percent capping for consensus versus anti-consensus is $\sim$2%. Considering positions −3 through +2, the difference, $\Delta$, in mean percent capping for consensus versus anti-consensus is $\sim$13% (Figure 3.11D).
Figure 3.11. CapZyme-Seq Analysis of NCIN Capping In Vivo. (A–D) Promoter sequence determinants for NCIN capping in vivo in *E. coli*. Percent capping histogram (A). TSS position histograms (B; mean ± SD of percentage of TSS at each position; n = 3). Relative percent capping difference distributions (C; the dashed line is 0, the solid lines are the means, consensus nucleotides are colored red, anti-consensus nucleotides are colored blue). Percent capping histograms for −3 through +2 consensus (red) and anti-consensus (blue) sequences for NCIN capping in vivo (D). (E) Identification and quantitation of NCIN-capped sRNAs in vivo in *E. coli*. Bold indicates sRNA sequences previously identified as NAD⁺ capped (Cahová et al. 2015). In the promoter sequences,
gray shading represents the promoter −10, extended −10, and −35 promoter elements, and colors indicate matches to the −3 through +2 consensus (red) and anti-consensus (blue) sequences for NCIN capping in vivo. Percent capping values represent the mean of three independent measurements. In the column for percent capping, the number reported previously in (Nübel, Sorgenfrei, and Jäschke 2017) for GcvB is in parentheses.

3.4.12 Identification and Quantitation of NCIN-Capped sRNAs In Vivo

Identities of several NAD⁺-capped sRNA and sRNA-like 5′-RNA fragments in *E. coli* total cellular RNA have been reported (Cahová et al. 2015). Here, we applied CapZyme-seq to identify and quantify NCIN capping of all annotated *E. coli* sRNAs. We isolated *E. coli* total cellular RNA and performed CapZyme-seq^NudC^ using primers for the cDNA synthesis step designed to target 77 annotated sRNAs of *E. coli* (Keseler et al. 2017). Analysis of uncapped, 5′-triphosphate RNA using Rpp identified 16 sRNAs arising from promoters having A:T at the TSS position (**Figure 3.11E**). Analysis of NCIN-capped RNA using NudC shows that all 16 sRNAs exhibit NCIN capping (**Figure 3.11E**), including eight previously reported as NAD⁺ capped (Cahová et al. 2015) (**Figure 3.11E**, text in bold). NCIN capping levels for the 16 sRNAs ranged from 1.6% to 22.4%. Three RNAs shows particularly high NCIN capping levels: SibE, SibD, and OxyS (22.4%, 21.3%, and 13.3%). SibE and SibD are anti-toxin sRNAs, and OxyS is an sRNA regulator of oxidative stress. We note that the two most highly capped sRNAs are transcribed from promoters that match the four most important positions, −2 to +2, of our consensus sequence for NCIN capping in vivo.
3.4.13  MASTER analysis of discriminator sequence determinants on TSS selection

To investigate whether there are sequence determinants for TSS selection outside the TSS region, we applied a next-generation-sequencing approach that enables comprehensive analysis of sequence determinants during transcription: MASTER (Vvedenskaya et al. 2015). MASTER entails generating transcripts from a library of barcoded randomized sequences and sequencing transcript ends (Vvedenskaya et al. 2015). To define effects on TSS selection of sequences outside the TSS region, we analyzed a template library containing all $4^{10}$ (~1,000,000) sequences at positions 1 to 10 bp downstream of the –10 element, extending the randomized sequence to include the “discriminator” (7–10), located between the TSS region and the –10 element (MASTER-N10; Figure 3.12A). Results of MASTER-N10 analysis reveal that the discriminator affects TSS selection (Figure 3.12, B to D). Changes in the discriminator change TSS selection by up to ~3 bp (Figure 3.12D) and change the mean TSS, averaged over the ~16,000 templates analyzed for each of the 64 discriminator sequences, by ~1 bp (Figure 3.12B). Discriminators having a purine at each position (RRR), particularly GGG, favor TSS selection at upstream-shifted positions, whereas discriminators having a pyrimidine at each position (YYY), particularly CCT, favor TSS selection at downstream-shifted positions (modal TSS for RRR, 7 bp downstream of –10 element; modal TSS for YYY, 8 bp downstream of –10 element; Figure 3.12, B and C). Results from MASTER-N10, where the discriminator is GTG, match results from MASTER-N7, where the discriminator is GTG, demonstrating the reproducibility of the approach (Figure 3.12, C and D). We conclude that the discriminator is a determinant of TSS selection.
A conserved region of transcription initiation factor σ, “σ region 1.2” (σ₁₂), makes sequence-specific protein-DNA interactions with the nontemplate strand of the discriminator in the transcription bubble in RP₀ (Haugen et al. 2006; Yu Zhang et al. 2012). These interactions confer specificity for GGG (Haugen et al. 2006; Yu Zhang et al. 2012; Feklistov et al. 2006). To determine whether sequence-specific σ₁₂-discriminator interactions affect TSS selection, we used MASTER-N10 to compare wild-type σ to a σ derivative having alanine substitutions that disrupt sequence-specific discriminator-σ₁₂ interactions: σ₁₂-mut (Haugen et al. 2006, 2008). The results show that disrupting σ₁₂-discriminator interactions markedly alters TSS selection for templates containing a GGG discriminator, resulting in a downstream shift in mean TSS (Figure 3.12E). We conclude that σ₁₂-discriminator interactions are a determinant of TSS selection.

The results in Figure 3.12 show that TSS selection can occur at any of five positions downstream of the –10 element—i.e., positions 6 to 10, with position 7 generally being preferred—and that discriminator sequence affects TSS selection. These results imply that RP₀ can accommodate ~17 Å (5 bp × 3.4 Å/bp) variation in the position of the TSS and that discriminator sequence affects the ability to accommodate this variation. It has been hypothesized that TSS selection at positions downstream of the modal TSS (generally position 7; Figure 3.12C) involves transcription-bubble expansion (“scrunching”), and TSS selection at positions upstream of the modal TSS involves transcription-bubble contraction (“anti-scrunching”) (Vvedenskaya et al. 2015; Robb et al. 2013). According to this hypothesis, RP₀ generally contains a 13-bp transcription bubble that places position 7 in the RNAP active-center initiating nucleoside triphosphate (NTP) site (“i site”) and position 8 in the RNAP active-center NTP addition site (“i+1 site”; TSS = 7). For TSS selection to occur at positions 8, 9, or 10, it is hypothesized that the
downstream DNA duplex is unwound by 1, 2, or 3 bp; the unwound DNA is pulled into and past the RNAP active center, and the unwound DNA is accommodated as bulges within the transcription bubble, yielding a scrunched complex (TSS = 8, 9, or 10). For TSS selection to occur at position 6, it is hypothesized that the opposite occurs: Downstream DNA is rewound by 1 bp, downstream DNA is extruded from the RNAP active center, and the extrusion of DNA is accommodated by stretching DNA within the transcription bubble, yielding an anti-scrunched complex (TSS = 6). Two lines of evidence support this hypothesis: Single-molecule fluorescence resonance energy transfer results suggest that transcription-bubble size in RP\textsubscript{O} can vary (Robb et al. 2013); and negative supercoiling, which provides a driving force for transcription-bubble expansion, favors TSS selection at downstream positions (Vvedenskaya et al. 2015). However, direct evidence for this hypothesis has not been reported.
Figure 3.12. Sequences upstream of TSS region affect TSS selection. (A) Promoter sequences analyzed in MASTER-N7 and MASTER-N10. Promoter –35, –10, and discriminator elements are indicated. Green, randomized nucleotides. (B) Effect of discriminator on position of TSS (numbered in base pairs downstream of –10 element). Data show means and 99.9% confidence intervals for each of the 64 discriminator sequences (~16,000 templates analyzed for each discriminator). Green, GGG and other RRR discriminators; blue, CCT and other YYY discriminators; red, GTG discriminator.
3.4.14 MASTER analysis of RNAP leading-edge and trailing-edge positions in transcription bubble

Transcription-bubble expansion (scrunching) occurs in initial transcription, where it is coupled to RNA synthesis (Winkelman et al. 2015; Kapanidis et al. 2006; Revyakin et al. 2006). A hallmark of scrunching during initial transcription is that the RNAP trailing edge remains stationary relative to DNA, whereas the RNAP leading edge moves relative to DNA (Winkelman et al. 2015; Kapanidis et al. 2006; Revyakin et al. 2006). Here, we investigated whether this hallmark of scrunching is a feature of TSS selection. We used unnatural amino acid–mediated protein-DNA photo–cross-linking to define the RNAP trailing-edge position and RNAP leading-edge position in RP₀ on a MASTER-N10 library (MASTER-N10-XL; Figure 3.13A). To perform MASTER-N10-XL, we incorporated the photoactivatable unnatural amino acid p-benzoyl-L-phenylalanine (Bpa) (Chin et al. 2002; Winkelman et al. 2015) at specific sites at the RNAP trailing edge and RNAP leading edge; formed RP₀ between the Bpa-containing RNAP derivatives and the MASTER-N10 library; and irradiated complexes with ultraviolet light to induce cross-linking between Bpa and adjacent DNA nucleotides (Figure 3.13A). We then mapped
positions of cross-links on each of the 4^10 sequences by primer extension and high-throughput sequencing (Figure 3.13A). The method defines the mean RNAP trailing-edge position, the mean RNAP leading-edge position, and the mean distance between them (RNAP trailing-edge/leading-edge distance; Figure 3.13).

The results show that changes in the discriminator change the RNAP trailing-edge/leading-edge distance in the same manner that changes in the discriminator change TSS selection (Figure 3.12 and Figure 3.13). Changes in the discriminator change the RNAP trailing-edge/leading-edge distance by up to ~3 bp and change the RNAP trailing-edge/leading-edge distance, averaged over the ~16,000 templates analyzed for each of the 64 discriminator sequences, by ~1 bp (Figure 3.13B). The RNAP trailing-edge/leading-edge distance is shortest for RRR, especially GGG, discriminators and longest for YYY, especially CCT, discriminators (Figure 3.13, B to D). Disruption of σ₁₂-discriminator interactions results in a marked increase in RNAP trailing-edge/leading-edge distance for templates containing a GGG discriminator (Figure 3.13E).

We next compared the effects of discriminator sequence on TSS selection (Figure 3.12) to effects of discriminator sequence on RNAP trailing-edge position and RNAP leading-edge position (Figure 3.13 and Figure 3.14). The results show that, as the position of the TSS changes by 1 bp, the RNAP leading-edge position changes by 1 bp (Figure 3.14A), but the RNAP trailing-edge position does not change (Figure 3.14A). Thus, TSS selection exhibits a defining hallmark of scrunching: namely, the RNAP leading edge moves, whereas the RNAP trailing edge does not move (Figure 3.14B). The results provide support for the hypothesis that flexibility in TSS selection is mediated by
scrunching/anti-scrunching, and that the discriminator affects TSS selection by modulating the extent of scrunching/anti-scrunching.
Figure 3.13. Sequences upstream of TSS region affect RNAP trailing-edge/leading-edge distance. (A) MASTER-N10-XL. (B) Effect of discriminator on RNAP trailing-edge/leading-edge distance in RPo (symbols as in Figure 12). (C) RNAP trailing-edge/leading-edge distances. (D) Decreases and increases in RNAP trailing-edge/leading-edge distance with the ~16,000 GGG- and ~16,000 CCT-discriminator templates (green and blue, respectively) relative to the ~16,000 GTG-discriminator templates (red). Dashed lines indicate mean trailing-edge/leading-edge distances. (E) Effect of $\sigma_{1.2}$-discriminator interactions on RNAP trailing-edge/leading-edge distance (increase in RNAP trailing-edge/leading-edge distance for ~16,000 GGG-discriminator templates on replacement of $\sigma$ by $\sigma_{1.2}$ mutant).
Figure 3.14. As TSS changes, RNAP leading-edge position changes, but RNAP trailing-edge position does not change. (A) RNAP trailing-edge position (left; slope ~0) and RNAP leading-edge position (right; slope ~1) as a function of mean TSS for each of the 64 discriminator sequences (~16,000 templates analyzed for each discriminator; green, RRR discriminators; blue, YYY discriminators). (B) Interpretation: Changes in TSS selection result from changes in DNA scrunching. Gray, RNAP; yellow, σ; blue, –10 element nucleotides; purple, discriminator nucleotides; i and i+1, NTP binding sites; red, Bpa and nucleotide cross-linked to Bpa; boxes, DNA nucleotides (nontemplate-strand...
nucleotides above template-strand nucleotides; nucleotides downstream of –10 element numbered). Scrunching is indicated by bulged-out nucleotides. Anti-scrunching is indicated by a “stretched” nucleotide-nucleotide linkage.

3.5 Discussion

We systematically profiled the effects of promoter region sequence variants on a series of biochemical properties in transcription initiation, including TSS selection, slippage synthesis, transcript yield, and NCIN capping. Also, we found that DNA scrunching by RNAP is a mechanism for TSS selection prior to RNA synthesis, from the comprehensive profiling of the RNAP leading- and trailing-edge positions on different discriminator and promoter region sequence contexts. To enable these massive parallel profilings of RNAP-DNA interactions, we have developed three HTS applications, including MASTER, CapZyme-seq, and mNET-seq. We propose that these HTS applications can be adapted to systematically investigate other types of protein-nucleic acid interactions.

3.5.1 MASTER analysis provides a comprehensive description of transcription initiation

We documented three measurable outputs of transcription initiation—TSS position, productive slippage synthesis, and yields of full-length transcripts—for a library comprising $4^7$ (~16,000) sequence variants of a consensus E. coli promoter. The results define full inventories of transcription start sites (“TSSomes”) of E. coli RNAP in vitro and in vivo and full inventories of transcripts generated by productive slippage synthesis (“slippomes”) of E. coli RNAP in vitro and in vivo. Our analyses of productive slippage synthesis indicate that slippage synthesis occurs from the majority of TSS-region DNA
sequences and reveal slippage by increments of two nucleotides occurs at surprisingly high levels (Figure 3.4).

MASTER analysis of the effects of TSS-region sequences on yields of full-length transcripts shows that TSS-region DNA sequences can have profound, up to 100-fold, effects on transcript yield (Figure 3.5). Furthermore, the impact of TSS-region sequence on the range of expression varies in response to changes in NTP concentrations. Our findings that promoter TSS-region sequences can dictate a wide range of expression levels suggests that these sequences serve as a reservoir of expression level diversity that should easily be accessible to mutational processes and natural selection for tuning or altering promoter output.

3.5.2 MASTER as a new approach for comprehensive analysis of the relationship between nucleic-acid sequence and functional output

Analyses of the behavior of chromosomally encoded promoters using high-throughput approaches (e.g., RNA-seq, ChIP-seq) have provided a wealth of information regarding mechanisms employed by cells to regulate gene expression. However, every promoter is an evolved, unique sequence, which constrains the ability of researchers using such approaches to infer universal properties based on aggregate behavior. Furthermore, for sequence regions of more than a few bases, the total sequence diversity represented by all promoters in a genome is significantly less than the maximum theoretical diversity contained in an equivalent length of randomized DNA sequence. MASTER overcomes the inherent limitations imposed by analysis of chromosomally encoded promoters by enabling the comprehensive measurement of transcription output for all possible
sequence variants of a given region of a transcription unit. In addition, MASTER allows the analysis of the behavior of all possible sequence variants in a region of interest to be performed over diverse conditions in vitro and in vivo.

Furthermore, although for sequence-specific promoter elements one can predict the effect of mutations on the overall affinity of RNAP for the promoter element, correlating the effect on transcription output is not straightforward. Thus, predicting how a given promoter will respond to alterations in conditions and identifying the sequence determinants that dictate the response represents an immense challenge. MASTER overcomes these limitations by enabling systematic variation of sequence attributes in a controlled fashion. Thus, we anticipate that the results obtained from these and future studies using MASTER will enable more accurate predictions of the behavior of chromosomally encoded promoters and will inform the design of synthetic promoters for use in artificial biological circuits. In addition, MASTER can be readily adapted for comprehensive analysis of sequence determinants for transcription elongation, transcription termination, or any other biological process that involves nucleic-acid interactions.

3.5.3 Sequence-Specific RNAP–CRE Interactions in TSS Selection

We found that sequence-specific interactions between RNAP and the downstream segment of the nontemplate strand of the transcription bubble (CRE) are a determinant of TSS selection. In particular, using 5’ mNET-seq, we define a role of sequence-specific recognition of a G at the most downstream position of the CRE ($G_{CRE}$) during TSS selection in the context of a library of $4^7$ (~16,000) TSS region sequences of a consensus core promoter in vitro and in vivo and in the context of natural promoters in *E. coli* in vivo (Figure 3.6). The results indicate that TSS selection is a multifactorial
process, in which the ultimate outcome for a given promoter reflects the contributions of multiple promoter sequence determinants and multiple reaction conditions. Because sequence-specific RNAP–CRE interactions are only one of several determinants of TSS selection, their quantitative significance at different promoters differs. At some promoters, such as PsecE and PhemC, sequence-specific RNAP–CRE interactions have quantitatively large, ≥20%, effects on TSS selection (Figure 3.6C), whereas at other promoters, the quantitative effects of RNAP–CRE interactions are smaller.

3.5.4 Proposed Basis for Promoter Sequence Specificity for NCIN Capping with NAD$^+$

The results in Figure 3.8, Figure 3.9, and Figure 3.10 show that the efficiency of NCIN capping with NAD$^+$ is determined by the sequence at the TSS (+1), the sequence at the three positions immediately upstream of the TSS (−3 to −1), and the sequence at the three positions immediately downstream of the TSS (+2 to +4). There is an essentially absolute preference for A at the TSS (Figure 3.8B). At the first position upstream of the TSS, position −1, the sequence has very strong effects on efficiency of NAD$^+$ capping (up to at least 16-fold; Figure 3.9B). At the second and third positions upstream of the TSS, positions −2 and −3, the sequence has small but significant effects (up to at least 2.7-fold and 2.2-fold, respectively; Figure 3.9B). At the first position downstream of the TSS, position +2, the sequence has large effects (up to at least 6.8-fold; Figure 3.9B). At the next two positions downstream of the TSS, positions +3 and +4, the promoter sequence has small but significant effects (up to at least 2.9-fold and 2.7-fold, respectively; Figure 3.9B). The essentially absolute preference for an A:T base pair at the TSS, position +1 results from the Watson-Crick base-pairing preference of the adenosine moiety of NAD$^+$ with a T on the template strand.
Structural modeling suggests that the very strong preference for R (Y on template strand) at the position immediately upstream of the TSS, position −1, can be understood in terms of pseudo-base-pairing of the NAD\(^+\) nicotinamide moiety with the DNA template-strand base at position −1. The NAD\(^+\) nicotinamide can be positioned to form a nicotinamide:Y pseudo-base pair with template strand C at position −1 or, with a 180° rotation about the pyridine-amide bond of the NAD\(^+\) nicotinamide, with template-strand T at position −1, in each case, forming two H-bonds with Watson-Crick H-bonding atoms of the template strand and stacking on the NAD\(^+\) adenine base. In contrast, the NAD\(^+\) nicotinamide moiety would experience severe steric clash with template strand A or G at position −1.

Structural modeling suggests that the specificity for R (Y on template strand) at position −2 also can be understood in terms of stacking preferences for pseudo-base-pairing by the NAD\(^+\) nicotinamide moiety to the template-strand base at position −1. A template-strand Y at position −2 can stack favorably on the NAD\(^+\) nicotinamide moiety of a nicotinamide:Y pseudo-base pair, but a template-strand R at position −2 would clash with the NAD\(^+\) nicotinamide moiety.

The strong specificity for S (S on template strand) at the first position downstream of the TSS, position +2, can be understood in terms of differences of 1/Km for the incoming extending NTP, which base pairs with the template-strand base at position +2 (Jensen et al. 1986; Rhodes and Chamberlin 1974), together with different sensitivities to this parameter of NAD\(^+\)-mediated initiation to ATP-mediated initiation.
The specificity for W:W base pairs at positions +3 and +4 observed in vitro potentially can be understood in terms of differences in DNA duplex stabilities and corresponding DNA unwinding energies for W:W base pairs versus S:S base pairs, together with different sensitivities to these parameters of NAD⁺-mediated initiation to ATP-mediated initiation.

3.5.5 Scrunching in transcription start site selection

The results of MASTER-XL analysis indicate that the flexibility in TSS selection occurs through changes in scrunching/anti-scrunching in RP₀. We propose that RP₀ uses thermally driven DNA fluctuations to access an ensemble of transcription-bubble sizes. Transcription-bubble expansion (scrunching) places downstream DNA in contact with the RNAP active center, facilitating downstream TSS selection; and transcription-bubble contraction (anti-scrunching) places upstream DNA in contact with the RNAP active center, facilitating upstream TSS selection. According to this model, the discriminator alters TSS selection by altering the energy landscape describing the ensemble of transcription-bubble sizes in RP₀. The scrunching that occurs in TSS selection is mechanistically analogous to the scrunching that occurs during initial transcription (Revyakin et al. 2006; Winkelman et al. 2015; Kapanidis et al. 2006). However, scrunching in TSS selection occurs before NTP binding and nucleotide incorporation and, in the absence of an additional energy source, is driven by energy available from the thermal bath and therefore limited to ~1 to 3 bp—rather than occurring after NTP binding and nucleotide incorporation, being driven by a combination of thermal energy, NTP binding, and nucleotide incorporation, and being able to span tens of base pairs. We suggest that, in the presence of an additional energy source, scrunching in TSS selection could access a larger range of TSS positions, and, in particular, we speculate that this occurs with transcription factor IIH (TFIIH)—dependent adenosine 5'
triphosphate (ATP) hydrolysis as the additional energy source in the long-range “TSS scanning” observed with eukaryotic RNAP II in some species (Fazal et al. 2015).
4 Scedar: a scalable Python package for single-cell RNA-seq exploratory data analysis

Abstract: In single-cell RNA-seq (scRNA-seq) experiments, the number of individual cells has increased exponentially, and the sequencing depth of each cell has decreased significantly. As a result, analyzing scRNA-seq data requires extensive considerations of program efficiency and method selection. In order to reduce the complexity of scRNA-seq data analysis, we present scedar, a scalable Python package for scRNA-seq exploratory data analysis. The package provides a convenient and reliable interface for performing visualization, imputation of gene dropouts, detection of rare transcriptomic profiles, and clustering on large-scale scRNA-seq datasets. The analytical methods are efficient, and they also do not assume that the data follow certain statistical distributions. The package is extensible and modular, which would facilitate the further development of functionalities for future requirements. The open source package is distributed under the terms of the MIT license at https://pypi.org/project/scedar.

4.1 Table of Contents

- Introduction
- Methods
- Results
- Discussion
4.2 Introduction

Cost-effective large-scale transcriptomic profiling of individual cells is enabled by the development of microfluidic, nanodroplet, and massively parallel sequencing technologies. Using these technologies, single-cell RNA-seq (scRNA-seq) experiments usually generate transcriptomic profiles of thousands to millions of individual cells (Svensson, Vento-Tormo, and Teichmann 2018). Therefore, scRNA-seq has become more commonly used to either study specific biological questions or comprehensively profile certain tissues or organisms (Filbin et al. 2018; Cao et al. 2017; Regev et al. 2017).

The large number of cells in scRNA-seq datasets require the data analysis programs to be efficient. During computational analysis, data and results are stored in the virtual memory of operating system. If the utilized virtual memory greatly exceeds the amount of physical memory, the data and results will be shuffled between hard drive and physical memory, which would significantly slow down the computation by 6 to 100,000 times depending on the specific data access pattern (Jacobs 2009). Therefore, the computational program needs to avoid unnecessary copies throughout the analysis, when analyzing millions of single cell transcriptomic profiles. Also, the runtime of most data analysis methods increases with the number of single cells, which may take several days to complete the analysis of a large scRNA-seq dataset, so the computational implementation of the methods needs to exploit parallel processing when possible to reduce the runtime. Furthermore, the computational program need to support efficient debugging, because each trial may take a long time to reach the error. The error messages need to be informative for the users to locate and fix the errors. The interruptions need to be readily handled to stop the execution of an unnecessary step.
The exceptions and errors need to be handled gracefully to prevent the system from crashing, so that the results in previous analysis can still be used afterwards.

The statistical methods for scRNA-seq data analysis need to handle high dimensionality, low signal-to-noise ratio, and different statistical characteristics of data generated from different scRNA-seq platforms and protocols (Kiselev, Andrews, and Hemberg 2019; Ziegenhain et al. 2017; Dueck et al. 2016). The number of dimensions in scRNA-seq data is proportional to the number of annotated genes, which could reach 60,000 (Frankish et al. 2019). Under such high dimensionality, the data points, i.e. single cells transcriptomic profiles in scRNA-seq, become similar to each other, even when all dimensions are meaningful (Domingos 2012). This property of high-dimensional data is one example of the “curse of dimensionality” (Domingos 2012), which makes it difficult to identify rare cell populations in scRNA-seq datasets (Aggarwal and Yu 2001), such as cancer stem cells and transient cell states during development that are not extensively characterized in previous research. The identification of rare cell populations in scRNA-seq datasets is further complicated by low signal-to-noise ratio, which is represented by high gene-dropout rate and low read counts per cell. The high gene-dropout rate refers to the observation that a large fraction of cells have read counts equal to 0 in certain genes while other cells have read counts greater than 0, even when the cells belong to the same cell type (Zappia, Phipson, and Oshlack 2017). As the total number of reads per cell is low, which is centered around a thousand in some scRNA-seq datasets (Macosko et al. 2015; Cao et al. 2019; Han et al. 2018), noise in the read counts, such as stochastically gaining or missing some reads in certain genes, would have non-trivial impact on the results. Furthermore, scRNA-seq data generated from different platforms and protocols have distinct statistical characteristics, such as the distribution of the total read counts per cell, the distribution of the gene read counts within a single cell, and the
distribution of read counts of a gene across all cells. Therefore, the parameters of analytical methods need to be optimized for different types of scRNA-seq data through extensive exploratory data analysis.

In order to facilitate the exploration of large-scale scRNA-seq datasets, we developed a scalable and reliable Python package, scedar (single-cell exploratory data analysis for RNA-seq) (Yuanchao Zhang and Taylor 2019). Scedar provides analytical routines for visualization, gene dropout imputation, rare transcriptomic profile detection, clustering, and identification of cluster separating genes. The visualization methods are integrated with the efficient scRNA-seq data structures to provide intuitive, convenient, and flexible plotting interfaces. We implemented methods to impute gene dropouts and detect rare transcriptomic profiles based on the k-nearest neighbor (KNN) algorithm. For clustering analysis, we provide a novel cell clustering algorithm named MIRAC, minimum description length (MDL) iteratively regularized agglomerative clustering. In order to identify genes that are able to distinguish the clusters, we provide a method using a sparsity-aware gradient boosted tree system, XGBoost (T. Chen and Guestrin 2016).

Scedar showed efficient runtime and memory usage on multiple simulated and real scRNA-seq datasets. The runtime is reduced by caching and parallel processing, which exploits modern high-performance computing architectures that have large amount of memory and multiple central processing unit (CPU) cores. The memory usage is optimized by lazy-loading and object-referencing. Lazy-loading is a software development technique to defer the creation of data-structures in memory until they are used by the analysis. Object-referencing enables the sharing of the available data-structures without extra copying. Furthermore, scedar can be applied to datasets generated from different scRNA-seq platforms, because we do not make specific
assumptions on statistical distributions of the datasets. Instead, we incorporate efficient implementations of machine learning methods to facilitate extensive data exploration, through which the statistical properties of the data could be observed and used to guide the selection of appropriate methods, from more than 275 available ones, for further analysis (Zappia, Phipson, and Oshlack 2018).

4.3 Methods

4.3.1 scedar package design

We designed scedar in an object-oriented manner for quickly exploring large-scale scRNA-seq transcription level matrices on a remote server utilizing parallel computing techniques, in order to provide a robust and extensible platform for EDA, rather than surpassing the analytical performance of any of ≥ 275 existing scRNA-seq data analysis methods (Zappia, Phipson, and Oshlack 2018). The application programming interface (API) is designed to be intuitive to users familiar with the R programming language. The standard analysis workflow is to explore the dataset, cluster cells, and identify cluster separating genes (Figure 4.1), which is implemented as four main modules: data structure, KNN, clustering and visualization.

The core data structure stores each transcription level matrix as a standalone instance, and it can easily be extended to support customized analytical procedures. The built-in extensions include common procedures like pairwise distance computation, Principal Component Analysis (PCA), t-SNE (Maaten and Hinton 2008), UMAP (McInnes and Healy 2018), and k-nearest neighbor graph (Jacomy et al. 2014). We optimized time and
memory efficiency with the following design patterns: parallel computing, lazy loading, caching and copy-on-write.

The KNN and clustering modules utilize the data structure and parallel computing to efficiently perform analytical procedures, and the results are stored in the data structure for further reference.

The visualization module contains plotting methods optimized for large datasets, especially for plotting heatmaps. For example, it takes less than two minutes to generate a heatmap image from a 50,000 x 20,000 matrix containing random standard normal entries.

Preprocessing is implemented as selection and transformation routines of the core data structure, which is not a focus of scedar. The package is designed to identify the necessity of certain preprocessing procedures by extensively exploring the original state of the data. Although preprocessing, such as batch effect correction and normalization, could facilitate the detection of biological variances between cells, applying preprocessing methods correctly requires careful validation of their assumptions, otherwise they may introduce unwanted bias or variability (Hicks and Irizarry 2015).

4.3.2 Minimum description length iteratively regulated agglomerative clustering

Minimum description length (MDL) iteratively regulated agglomerative clustering (MIRAC) extends hierarchical agglomerative clustering (HAC) (Müllner 2011) in a divide and conquer manner for scRNA-seq data. Input with raw or dimensionality reduced
scRNA-seq data, MIRAC starts with one round of bottom-up HAC to build a tree with optimal linear leaf ordering (Bar-Joseph, Gifford, and Jaakkola 2001), and the tree is then divided into small sub-clusters, which are further merged iteratively into clusters. Because each individual cluster becomes more homogenous with higher number of clusters, the iterative merging process is regularized with the MDL principle (Hansen and Yu 2001). The asymptotic time complexity of MIRAC algorithm is $O(n^4 + mn^2)$, where $n$ is the number of samples, and $m$ is the number of features. The space complexity is $O(n^2 + mn)$. Relevant mathematical theories and notations of MDL are briefly described in the following section. The pseudo-code of MIRAC is shown in Algorithm 1.
Algorithm 1: Minimum description length iteratively regulated agglomerative clustering
Comparing to HAC, MIRAC is not designed to be faster but rather to improve the cluster robustness. The asymptotic time complexity of MIRAC is the same as HAC with optimal leaf ordering. However, the use of MDL rather than deterministic similarity metrics, improves the noise tolerance by estimating similarity with probabilistic models to give more weight on signal and less weight on noise, assuming that the signal is stronger than noise.

4.3.2.1 Rationale

The rationale behind MIRAC is to reduce the number of cell partitions that need to be evaluated and find a good one among them.

The number of all possible partitions of a set with \( n \) elements is the Bell number, which could be computed with the following recurrence equation:

\[
B_n = \sum_{k=0}^{n-1} \binom{n-1}{k} B_k,
\]

where \( n \geq 1 \) and \( B_0 = 1 \) (Wilf 2005). It is computationally intractable to compute the code lengths of \( B_n \) partitions, so we reduced the number of cell partitions to evaluate with the following steps:

1. We only evaluate the partitions of \( n \) cells that consecutively divide the optimal HAC tree \( T \) leaf ordering. The HAC tree with optimal leaf ordering maximizes the sum of similarity measurements between adjacent items in the leaf ordering (Bar-Joseph, Gifford, and Jaakkola 2001), which could be computed with an algorithm developed by Bar-Joseph et al. that runs in time \( O(n^4) \). The number of partitions that consecutively divide an ordering is the same as the number of compositions
of \( n \), which is \( C_n = 2^{n-1} \). Because this number still grows exponentially with regard to \( n \), it is necessary to further reduce the set of partitions for evaluation.

2. Within step 1 partitions, we only evaluate those with all clusters having \( \geq t \) cells. The number of such partitions is the same as the compositions of \( n \) with all summands \( \geq t \), which is

\[
C_n^{\{t,t+1,\ldots,n\}} = \sum_{k=1}^{[n/t]} \binom{n - k(t-1) - 1}{k-1}
\]

given by (Abramson 1976). The growth rate of \( C_n^{\{t,t+1,\ldots,n\}} \) with regard to \( n \) is smaller, but we still need to reduce it further. For example, \( C_n^{\{20,21,\ldots,n\}} \) for \( n \) in an \( \langle 20, 40, 60, 80, 100, 120, 140, 160, 180, 200 \rangle \) are \( \langle 1, 2, 23, 274, 2695, 24941, 232016, 20628613, 194570810 \rangle \), and the values of \( 2^n \) are \( \langle 1048576, 1099511627776, \ldots, 1.607 \times 10^{60} \rangle \).

3. Within step 2 partitions, we only evaluate the ones that could be generated by merging adjacent clusters of a specific partition \( P^s = \{P_1^s, P_2^s, \ldots, P_k^s\} \), where \( P^s \) is generated by recursively divide the root HAC tree \( \bar{T} \) until the partitioned subtree has \( \leq t - 1 \) leaves. Thus, \( \lceil n/(t-1) \rceil \leq k \leq n \). Let \( M(P^s) \) denote the number of step 2 partitions that could be generated by merging adjacent clusters of \( P^s \). The upper bound of \( M(P^s) \) is the same as the number of step 2 partitions, which could be reached when \( k = n \). The lower bound of \( M(P^s) \) is not straightforward, since the merged partitions should have all clusters with \( \geq t \) cells.

In order to find a good cell partition \( P^g \) in the subset of all possible ones, we iteratively inspect each cluster of \( P^s \) and merge it with either its left or right adjacent cluster.
according to the similarity determined by the two-stage MDL scheme described in the following sections. The procedure has the following steps:

1. Merge $P^s$ clusters in the beginning of the optimal ordering until the merged set contains $\geq t$ cells (line 9 Algorithm 1).

2. Similarly, merge $P^s$ clusters at the end of the optimal ordering until the merged set contains $\geq t$ cells (line 11 Algorithm 1).

3. For every cluster $P^s_i$ in the middle, determine the similarity between $P^s_i$ and the cluster on the left and right, and merge $P^s_i$ with the more similar cluster (line 20 - 26 Algorithm 1). Although the cluster on the left $P^s_{i-1}$ always has $\geq n_{\text{cluster}}^{\text{min}}$ cells, the cluster on the right $P^s_{i+1}$ may have a minimum of one cell, so that the determined similarity between $P^s_i$ and $P^s_{i+1}$ is sensitive to noise. In order to improve the robustness of similarity comparison, instead of determining the similarity between $P^s_i$ and $P^s_{i+1}$ we determine the similarity between $P^s_i$ and $P^s_{i+m}$ (line 20 Algorithm 1), where $|P^s_{\text{rmm}|} \geq n_{\text{cluster}}^{\text{min}}$, $P^s_{\text{rmm}|} - P^s_{i+m} < n_{\text{cluster}}^{\text{min}}$, and $\text{rmm}$ is the shorthand for right minimax.

4. Once $|P^s_i| \geq n_{\text{cluster}}^{\text{min}}$, determine the similarity between $P^s_{i-1}$ and $P^s_i$, and merge them if their similarity is above a certain threshold, otherwise split them (line 32 - 36 Algorithm 1). The code length of merged $X_{\text{list}}(P^s_{i-1} \cup P^s_i)$ is

$$L_m = L(X_{\text{list}}(P^s_{i-1} \cup P^s_i)) .$$

The code length of divided $X_{\text{list}}(P^s_{i-1} \cup P^s_i)$ is

$$L_d = L(X_{\text{list}}(P^s_{i-1} \cup P^s_i), \{P^s_{i-1}, P^s_i\}) .$$

If $L_d - L_m \geq -\log_{\text{min}}(\text{abs}(L_m))$, merge $P^s_{i-1}$ and $P^s_i$, otherwise split them. This conditional statement generalizes the situations where $L_m$ is either non-negative or negative.
5. Continue inspecting the next cluster in $P^S$.
6. After inspecting all middle clusters, $P^S$ is the final updated $P^S$.

We use a standard HAC tree to perform MIRAC when the number of samples is larger than ten thousand, which usually results in decreased but still acceptable performances (Figure 4.2 and Figure 4.3). Although the optimal leaf ordering of the HAC tree is an important assumption, its computation takes too long when the number of samples is large. For example, it takes more than a week to compute the optimal leaf ordering of a dataset with about 68,000 samples. However, the time complexity of computing a good HAC linear ordering could be significantly reduced by implementing other ordering techniques (Aydin, Bateni, and Mirrokni 2016).

4.3.2.2 Analysis of complexity

Complexity analysis is used to evaluate the time and data storage space for an algorithm to complete on a computer, which is described as the growth function of running time and space as the input data size increases using big O notation (Leiserson et al. 2001). The big O notation formally describes a type of function without specific coefficients, which is reviewed by Graham et al. in details (Graham et al. 1989).

The time complexity of MIRAC is $O(n^4 + mn^2)$, where $n, m \geq 1$. The time complexity of computing the HAC tree with optimal leaf ordering is $O(n^4)$ (Bar-Joseph, Gifford, and Jaakkola 2001). The time complexity of finding $P^S$ is $O(mm^2)$, which is briefly analyzed as following.
The time complexity is $O(nm)$ for computing the code length $L(X_{n \times m}, P)$. In computing $L(X_{n \times m}, P)$, we compute the code lengths of each cluster and their labels. Since it takes $O(n)$ time to compute the code length of $n$ observations with a single feature, computing the code length of all clusters and features individually takes $O(nm)$ time, and computing the code length of $n$ labels take $O(n)$ time.

Similarly, the time complexity is $O(n_1 m + n_2 m)$ for computing the code length of $X_{n_1 \times m}$ encoded by a model fitted with $X_{n_2 \times m}$. Fitting a model on $X_{n_2 \times m}$ takes $O(n_2 m)$ amount of time. Using the model to encode $X_{n_1 \times m}$ takes $O(n_1 m)$ amount of time.

The time complexity of searching for $p^g$ is $O(mn^2)$. In the worst case scenario, $|P^g| = n$, and the inspecting cluster is always merged with the left hand side cluster in step 4. However, the impact of $n_{\text{cluster}}$ and step 3 on the time complexity is not straight forward, so we divide step 3 code length computation into three parts and analyze the upper bound of them individually. The divided three parts are the left cluster, inspecting cluster, and right minimax cluster. Because we keep increasing the size of the left cluster, the upper bound of computing the left cluster code length throughout the execution is $O(mn^2)$. The maximum size of the inspecting cluster is $n_{\text{cluster}} - 1$, so the upper bound of computing the inspecting cluster code length is also $O(mn^2)$ when every middle singleton is evaluated $n_{\text{min}} - 1$ times before merging with the left. The maximum size of the right minimax cluster is $2n_{\text{min}} - 1$, so the upper bound of evaluating right minimax cluster is also $O(mn^2)$, when every increment of the left cluster takes $n_{\text{min}}$ times of evaluating right minimax cluster. Summarizing these three parts, the overall upper bound is $O(mn^2)$. 
The space complexity of MIRAC is $O(nm + n^2)$, which could be decomposed into the following three parts. The space complexity of storing the $n \times m$ data matrix $X$ is $O(nm)$. The space complexity of storing the HAC tree with optimal leaf ordering is $O(n)$. The space complexity of storing the pairwise distance matrix is $O(n^2)$.

4.3.3 Minimum description length

Minimum description length (MDL) is the minimum size of information required to describe a set of data by a model (Hansen and Yu 2001). If the model plainly describes the data verbatim, the MDL of the data is equivalent to the size of the data. The MDL of the data can be reduced by a more sophisticated model that exploits the statistical properties in the data. For example, if 95% of the entries in a 10,000 x 10,000 matrix are 0s, a model can record only the indices of non-0 entries and keep a note that all other entries are 0s, which would be able to greatly reduce the MDL of the matrix. However, we do not want the model to become so complex that the information size for describing the model is very large. For example, a sequence of 10 single digit decimal integers can be described by a model that stores all possible sequences and refers each sequence by an index. Although the model is able to describe a dataset by a single index, the model needs to store 100 indices for all possible sequences, of which the size is even larger than describing the 10 integers plainly, so it is not a good model when evaluated by MDL framework. The principle of MDL is applied in statistics and machine learning to select the model that requires the smallest size of information to describe the model and data, and the practice is reviewed by Hansen and Yu in details (Hansen and Yu 2001).
We developed a cell clustering method for scRNA-seq data, MIRAC, based on the MDL principle (Hansen and Yu 2001), which finds the partition of cells that yields the shortest code length of the data. The input data of MIRAC could be any $n \times m$ matrix $X$ with $n$ cells and $m$ features, where the features could be any measurements containing information about the similarity between cells, e.g. the number of reads mapped to certain genes, distances to certain cells, or dimensionality reduced coordinates. In order to code the $n \times m$ data matrix $X$, we use a two-stage scheme (Hansen and Yu 2001), in which we code the partition of cells in the first stage and the partitioned data in the second.

4.3.3.1 Minimum description length in practice

Practically, let an observation $x$ of a random variable $X$ follow an arbitrary probability distribution $P$ with parameters $\Theta = \{\theta_1, \theta_2, \ldots, \theta_k\}$. If $P$ is continuous, let $f$ be its probability density function otherwise probability mass function. Then, the code length of $x$ is $-\log f(x)$ with an arbitrary base of 2 or the Euler's number $e$. In scedar, we consistently use $e$ as the base. The code length of $n$ observations $\{x_1, x_2, \ldots, x_n\}$ is the sum of the code lengths of individual observations, which gives an overall code length of

$$\sum_{i=1}^{n} -\log f(x_i)$$

. The code length of $P$ is the code length of $\Theta$ using uniform distribution for each parameter.

In a two-stage coding scheme, the overall code length of the data, i.e. the observations, is the sum of the following:

- Stage 1: code length of the statistical model $\mathcal{M} = \{P, \Theta\}$ (for the observations).
- Stage 2: code length of the observations encoded using $\mathcal{M}$. 
When there are multiple statistical models \( \{ M_1, M_2, \ldots, M_r \} \), we select the one that gives the shortest overall code length of the data. Intuitively, the closer the assumed distribution \( P \) to the true distribution of the observations, the shorter the overall code length of the observations. The simpler the assumed distribution \( P \), the shorter the code length of \( P \).

Importantly, the code length of observations encoded by a discrete model cannot be directly compared to the code length of the same observations encoded by a continuous model. For example, let \( X = [1, 1, 0, 1, 0] \) be our observations. The code length of \( X \) encoded by \( \text{Bernoulli}(0.6) \) is

\[
L(X, \text{Bernoulli}(0.6)) = -3 \log 0.6 - 2 \log 0.4 \approx 3.365.
\]

The code length of \( X \) encoded by \( \text{Uniform}(0, 1) \) is

\[
L(X, \text{Uniform}(0, 1)) = -3 \log 1 - 2 \log 1 = 0.
\]

Although \( \text{Bernoulli}(0.6) \) better describes \( X \) than \( \text{Uniform}(0, 1) \),

\[
L(X, \text{Bernoulli}(0.6)) > L(X, \text{Uniform}(0, 1)).
\]

The theoretical background of MDL is extensively reviewed by Hansen and Yu (Hansen and Yu 2001).

4.3.3.2 Two-stage coding scheme for clustered scRNA-seq data

In order to clarify the coding scheme for scRNA-seq clustering analysis, we introduce the following definitions and notations:

- Denote an ordered list \( Z \) of \( n \) items as \( \{ z_1, z_2, \ldots, z_n \} \).
- Define function \( \text{list}(\{ s_1, s_2, \ldots, s_n \}) = \{ s_1, s_2, \ldots, s_n \} \) to convert a set to a list.
- Let an \( n \times m \) matrix \( X \) be the data matrix of \( n \) cells and \( m \) features. We define the following operations:
  - \( X_{i, \cdot} \) gives the \( i \)th row of \( X \).
  - \( X_{\cdot, j} \) gives the \( j \)th column of \( X \).
  - \( X_{i,j} \) gives the entry of \( X \) at \( i \)th row and \( j \)th column.
  - \( X_{\{i_1,i_2,\ldots,i_r\},\{j_1,j_2,\ldots,j_c\}} \) gives a matrix of crossed entries of \( \{i_1,i_2,\ldots,i_r\} \) rows and \( \{j_1,j_2,\ldots,j_c\} \) columns in \( X \).

- Define a **partition** \( P \) of a set \( S \) as a set of non-empty subsets of \( S \) that are disjoint, of which the union is the same as \( S \). For example, \( \{\{1\},\{2,3\}\} \) is a partition of \( \{1,2,3\} \), whereas \( \{\{1\},\{2\}\} \) or \( \{\{\},\{1,2,3\}\} \) is not a partition of \( \{1,2,3\} \).

- Define operation \( |S| \) on any set \( S \) to give the number of elements in \( S \), i.e. cardinality of \( S \).

- We use a partition of a set of \( n \) different integers to denote a possible clustering result of \( n \) cells.

- For any partition \( P = \{P_1, P_2, \ldots, P_k\} \) of \( S \):
  - \( P \) is a **singleton partition** if \( k = 1 \).
  - We call \( P_i \) the \( i \)th cluster, where \( i \in \{1,2,\ldots,k\} \).
  - We define function \( I(i, P) \) on any element \( i \in S \), and \( I(i, P) = j \) such that \( i \in P_j \). Thus, we have a pair \( \langle i, I(i, P) \rangle \) for each element \( i \in S \), and we call \( I(i, P) \) as the **cluster label** of \( i \).

- Let \( \mathcal{B} \) be the list of cluster labels \( \langle I(1, P), I(2, P), \ldots, I(n, P) \rangle \).

For any \( X_{n\times m}, P = \{P_1, P_2, \ldots, P_k\} \), and cluster labels \( \mathcal{B} \), we encode \( X \) in the following two stages:
- Encode $B$ using categorical distribution. The code length of $B$ is

\[ L(B) = \sum_{i=1}^{k} -|P_i| \log \frac{|P_i|}{n}. \]

- Encode $X$ as $\{X_{\text{list}(p_1)}, \ldots, X_{\text{list}(p_k)}, \ldots, X_{\text{list}(p_t)}\}$. Within each row subset $X_{\text{list}(p_k)}$ of $X$, $m$ features are coded as individual random variables following arbitrary distributions. The code length of $X_{\text{list}(p_k)}$ is

\[ L(X_{\text{list}(p_k)}) = \sum_{j=1}^{m} \sum_{i \in P_k} - \log f_j(X_{i,j}), \]

where $f_j$ is the probability density or mass function of the assumed distribution of $X_{\text{list}(p_k),j}$.

We write the code length of $X$ with partition $P$ as $L(X, P)$. When $P$ is a singleton partition, we omit $P$ and write $L(X)$.

4.3.4 Mathematical theories on high-dimensional data analysis

The following two mathematical results on high-dimensional data analysis guided our development of analytical methods for scRNA-seq datasets.

4.3.4.1 Distances between points in high-dimensional space

As the number of features increases, all samples become closer in similarity metrics to each other (Domingos 2012), in a sense that that the distance between a sample and its nearest sample approaches to the distance between the sample and its farthest sample (Beyer et al. 1999; Aggarwal, Hinneburg, and Keim 2001). This property of distance in high-dimensional space is also called distance concentration effect (Zimek, Schubert, and Kriegel 2012). Therefore, analytical methods based on distances, such as
hierarchical agglomerative clustering, are less stable or, in other words, more susceptible to noise in the data. This result is mathematically described in the context of the nearest neighbors of a query point as following.

Definitions:

- Let any positive integer $m$ be the variable that the distance distributions may converge under. The variable $m$ can be interpreted as dimensionality, but this interpretation is not required by the proof of Theorem 1 given by Beyer et al. (Beyer et al. 1999).
- Let $n$ be the number of points.
- Let $X^m_1, X^m_2, \ldots, X^m_n$ be $n$ independent points such that $X^m_i \sim P^m_X$ for any $i \in \{1, 2, \ldots, n\}$, where $P^m_X$ a probability distribution.
- Let $Q^m$ be a query point sampled from the probability distribution $P^m_Q$ independently from $X^m_1, X^m_2, \ldots, X^m_n$.
- Let $0 < p < \infty$ be a constant.
- Define $D_m(X^m_i, Q^m)$ for any $i \in \{1, 2, \ldots, n\}$ as a function that returns a non-negative real number.
- Denote $D^\text{min}_m = \min\{D_m(X^m_i, Q^m) | i \in \{1, 2, \ldots, n\}\}$.
- Denote $D^\text{max}_m = \max\{D_m(X^m_i, Q^m) | i \in \{1, 2, \ldots, n\}\}$.

**Theorem 1. (Beyer et al.)** If

$$\lim_{m \to \infty} \text{var}\left(\frac{(D_m(X^m_1, Q^m))^p}{E[(D_m(X^m_1, Q^m))^p]}\right) = 0.$$ 

then for every $\epsilon > 0$

$$\lim_{m \to \infty} P[D^\text{max}_m \leq (1 + \epsilon)D^\text{min}_m] = 1.$$
The proof of Theorem 1 is given by Beyer et al. (Beyer et al. 1999). From the theorem, given that the distance distribution follows certain condition as $m$ increases, the distances of all points to the query point converges to a constant, which implies that the concept of nearest neighbor may not be meaningful (Beyer et al. 1999; Aggarwal, Hinneburg, and Keim 2001). The extent of restrictiveness of the precondition is also discussed by Beyer et al. (Beyer et al. 1999).

Although this property of distance between high-dimensional points affects analytical methods relying on distances (Beyer et al. 1999; Aggarwal, Hinneburg, and Keim 2001), the influences could be alleviated by dimensionality reduction, of which the performance in preserving the pairwise distances is characterized by the Johnson–Lindenstrauss lemma (Theorem 2) (Johnson and Lindenstrauss 1984).

4.3.4.2 Johnson–Lindenstrauss lemma

The Johnson–Lindenstrauss lemma generally states that $n$ high-dimensional points in Euclidean space can be embedded into a lower dimensional Euclidean space with $O(\log(n/\epsilon^2))$ dimensions for any $0 < \epsilon < 1$, while preserving the pairwise distances between $n$ points with errors within a factor of $\epsilon$ (Dasgupta and Gupta 1999; Johnson and Lindenstrauss 1984). The mathematical description of the theorem is summarized by Dasgupta and Gupta (Dasgupta and Gupta 1999) as the following:

**Theorem 2.** (Johnson–Lindenstrauss lemma) For any $0 < \epsilon < 1$ and any integer $n$, let $k$ be a positive integer such that

$$k \geq 4(\epsilon^2/2 - \epsilon^3/3)^{-1} \ln n.$$
Then for any set $V$ of $n$ points in $\mathbb{R}^d$, there is a map $f : \mathbb{R}^d \to \mathbb{R}^k$ such that for all $u, v \in V$,

$$(1 - \epsilon)u - v^2 \leq f(u) - f(v)^2 \leq (1 + \epsilon)u - v^2.$$  

Further this map can be found in randomized polynomial time.

The proof of the Johnson–Lindenstrauss lemma with elementary probabilistic techniques is given by Dasgupta and Gupta (Dasgupta and Gupta 1999).

4.3.5 Skewed root division of a hierarchical agglomerative clustering tree

We implemented a simple procedure to skew a hierarchical agglomerative clustering (HAC) before dividing the root into left and right subtrees (Figure 4.1). The skewed tree ensures that the smaller subtree of the root has $\geq n_{\text{cluster}} \cdot n_{\min}$ leaves, while preserves the ordering of leaves and maintains the invariants of a HAC tree. When $n_{\text{cluster}} \cdot n_{\min}$ is equal to half of the number of all leaves, the resulting division is similar to a balanced one.
Figure 4.1. Skewed division of hierarchical agglomerative clustering tree. Tree leaves are samples, which are marked by upper case letters. Tree inner nodes are agglomerated samples by arbitrary linkage, which are marked by number. The triangle under inner node 0 represents an arbitrary valid subtree with $\geq n_{\text{cluster}}^{\text{min}}$ leaves. The root division procedure divides a tree into left and right subtrees of the root node. The skewing procedure creates a minimum subtree of the root with $\geq n_{\text{cluster}}^{\text{min}}$ leaves, where $n_{\text{min}}^{\text{cluster}} = 3$ in this specific case.
Skewed root division is optional in Minimum description length (MDL) iteratively regulated agglomerative clustering (MIRAC). The procedure could be used to ensure that the divided sub-clusters are not too small, in order to improve the robustness of MDL estimation, because MDL estimation of a sub-cluster of too few samples is susceptible to noise.

4.3.6 scedar package development

Scedar is built upon various high-performance scientific computing and visualization packages. Scedar is also extensively benchmarked and tested by unit testing, with comprehensive coverage on statements and branches.

Scedar uses the following packages:

- numpy (Oliphant 2006) for matrix representation and operations.
- scipy (Virtanen et al. 2018) for fast Gaussian kernel density estimation, hierarchical clustering and sparse matrix.
- matplotlib (Hunter 2007) and seaborn (Waskom et al. 2017) for visualization.
- pandas (McKinney and Others 2010) for data frame representation.
- scikit-learn (Pedregosa et al. 2011) for parallel computation of pairwise distances, k-nearest neighbor data structure, PCA and t-SNE.
- XGBoost (T. Chen and Guestrin 2016) for scalable gradient boosting tree.
- networkx (Hagberg, Swart, and S Chult 2008) for graph data structure and visualization.
- ForceAtlas2 (Jacomy et al. 2014) for scalable force-directed graph layout.
We use the Python package pytest as the unit testing framework to ensure that scedar has expected behaviors. We tested each member of the package with multiple testing environments, in order to make sure that all statements and branches are executed in the tests, i.e. comprehensive code coverage. The code coverage is measured by the Python package coverage. Although comprehensive testing coverage does not guarantee that the package is bug-free, it eliminates obvious errors, e.g. accessing local variables before definition.

Comprehensive unit testing greatly helps with validating correctness, ensuring reproducibility and refactoring the code. We carefully tested our analytical procedures with multiple input datasets to cover standard and edge cases, in order to make sure that the results are correct and reproducible. We also confidently refactored the code multiple times throughout the development process to improve backend performance, accommodate special use cases, and reorganize intra-package dependencies. For a non-trivial package with multiple interrelated components like scedar, changes in certain components may unexpectedly affect other components that directly or indirectly use the changed ones, so that validating the correctness after the changes requires a significant amount of effort without comprehensive tests.

4.3.7 Cluster separating genes identification

We use XGBoost (T. Chen and Guestrin 2016), a scalable and sparsity-aware boosted tree system, to identify genes that are able to separate a specific set of clusters. This method is designed for data exploration after applying any one of a number of various statistical approaches that have been developed to identify differentially expressed genes (Vallejos, Marioni, and Richardson 2015; Kharchenko, Silberstein, and Scadden 2014; Qiu et al. 2017; Soneson and Robinson 2018; Korthauer et al. 2016), in order to
quickly identify genes or sets of genes that are able to separate an arbitrary set of clusters for further inspection.

Rather than providing a meticulous p-value for each gene among the compared clusters, we rank the genes by their importances on separating the clusters under comparison. The importance of a gene in the trained classifier is the number of times it has been used as an inner node of the decision trees. We use cross validation to train an XGBoost classifier on the compared clusters, and the classifier is essentially a bag of decision trees (T. Chen and Guestrin 2016). In order to alleviate the influences of stochasticity on interpretation, we use bootstrap with feature shuffling to better estimate the importance of genes in separating the compared clusters. The obtained list of important genes could be further explored by inspecting transcription level fold changes and decision tree structures.

Comparing to NSForest (Aevermann et al. 2018), a method based on random forest (Breiman 2001; Pedregosa et al. 2011) to identify a parsimonious set of cluster separating genes from scRNA-seq data, our method identifies all possible cluster separating genes using gradient boosting. Practically, NSForest version 1.3 is distributed on GitHub as a Python script without encapsulation or testing (checked on Oct 11, 2018), but our method is distributed through the Python Package Index, comprehensively tested, and easy to use through the user friendly API. With regard to scalability, our method uses a scalable implementation of gradient boosting algorithm (T. Chen and Guestrin 2016), whereas NSForest uses the implementation of random forest in scikit-learn (Pedregosa et al. 2011).
4.3.8 K-nearest neighbor methods

K-nearest neighbor analytical strategy exploits the similarity between data points for classification, regression, and imputation (Cover and Hart 2006; Bezdek, Chuah, and Leep 1986). In k-nearest neighbor methods, samples are considered as points in a space with each dimension representing a measured property, which is often referred to as a feature, of the samples. The similarity between samples can be evaluated with various distance metrics, which is extensively reviewed by Bellet et al. (Bellet, Habrard, and Sebban 2013). Generally, a distance metric is a function that takes two samples and output a numeric value to represent the distance between the two samples in their feature space. For example, the Euclidean distance metric is the following function,

$$d(p, q) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2},$$

where the $p$ and $q$ are two samples, and $q_i$ and $p_i$ are the data values on their $i$th dimension. The k-nearest neighbors of a sample are the k number of other samples that have the smallest distances from the sample. The k-nearest neighbors of a sample are informative, due to their similarity, to determine the category of the sample in classification, the relevant continuous property in regression, and the missing values in imputation.

We developed two methods based on the KNN algorithm to facilitate the exploration of scRNA-seq datasets. With relative large number of cells profiled in each scRNA-seq experiment, we assume that each one of the non-rare cells is similar to at least $k$ other cells in their transcriptomic profiles. With this assumption, we impute gene dropouts and detect rare transcriptomic profiles.
4.3.8.1 Impute gene dropouts

In an scRNA-seq experiment, if a truly expressed gene is not detected in a cell, the gene is considered a “dropout”, and such events are called gene dropouts (Kharchenko, Silberstein, and Scadden 2014). Gene dropouts may be caused by biological and technical reasons (Kharchenko, Silberstein, and Scadden 2014; Risso et al. 2018; Pierson and Yau 2015). The rationale behind the possible causes of biological dropouts mainly involves transcriptional bursting (Suter et al. 2011; Tantale et al. 2016) and RNA degradation. With regard to technical dropouts, the main concerns are the relatively small number of RNA transcripts of a gene, amplification efficiency, and batch effect (Zappia, Phipson, and Oshlack 2017).

We exploit the transcriptomic profiles of the k-nearest neighbors of a cell to impute the gene dropouts in the cell (Algorithm 2). The algorithm could take multiple iterations, so that the dropped-out genes that are expressed in all k-nearest neighbors could be imputed at first, and the ones that are expressed in most but not all of k-nearest neighbors could be imputed in the following iterations.
4.3.8.2 Detect rare transcriptomic profiles

We mark transcriptomic profiles as rare if they are distinct from their k-nearest neighbors, according to the pairwise similarity between cells (Algorithm 3). The
algorithm could take multiple iterations, so that the most distinct transcriptomic profiles could be marked at first and less distinct ones in the following iterations.

This method is provided mainly to facilitate detailed inspection of rare transcriptomic profiles rather than removing outliers from the data. Because rare transcriptomic profiles may have various biological and technical causes, samples and features in a dataset should only be removed after extensive exploratory data analysis and rigorous reasoning with domain specific knowledge. Closely comparing rare transcriptomic profiles with their nearest neighbors may also yield insights into their biological differences, which may further facilitate the identification of rare cell types and states.
4.3.9 Benchmark

We benchmarked the clustering and KNN performances of scedar on simulated and experimental scRNA-seq datasets. We obtained previously published experimental scRNA-seq datasets (Table 4.1). We also generated 50 simulated scRNA-seq read count datasets with Splatter (Zappia, Phipson, and Oshlack 2017). The simulation parameters are estimated by Splatter according to a Drop-seq dataset (Macosko et al.)
2015) and a 10x Genomics GemCode dataset (Zheng et al. 2017). Within each simulated dataset, the cells have 8 clusters taking approximately the following proportions: \(0.3, 0.2, 0.15, 0.15, 0.05, 0.05, 0.05, 0.05\), with a gene dropout rate around 5%.

**Table 4.1.** Real scRNA-seq datasets for benchmark

<table>
<thead>
<tr>
<th>Publication</th>
<th># cells</th>
<th># genes</th>
<th>Organism</th>
<th>Tissue</th>
<th>Protocol</th>
<th>Raw Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deng et al. (2014)</td>
<td>268</td>
<td>22,431</td>
<td>Mus musculus</td>
<td>embryo</td>
<td>Smart-Seq</td>
<td>read count</td>
</tr>
<tr>
<td>Pollen et al. (2014)</td>
<td>301</td>
<td>23,730</td>
<td>Homo sapiens</td>
<td>dermal, blood, pluripotent, and neural</td>
<td>SMARTer</td>
<td>TPM</td>
</tr>
<tr>
<td>Kolodziejczyk et al. (2015)</td>
<td>704</td>
<td>38,653</td>
<td>Mus musculus</td>
<td>Embryonic stem cell brain retina peripheral blood mononuclear cell</td>
<td>SMARTer</td>
<td>read count</td>
</tr>
<tr>
<td>Zeisel et al. (2015)</td>
<td>3005</td>
<td>19,972</td>
<td>Mus musculus</td>
<td>STRT-Seq UMI</td>
<td>read count</td>
<td></td>
</tr>
<tr>
<td>Macosko et al. (2015)</td>
<td>44,808</td>
<td>23,288</td>
<td>Mus musculus</td>
<td>Drop-seq</td>
<td>read count</td>
<td></td>
</tr>
<tr>
<td>Zheng et al. (2017)</td>
<td>68,579</td>
<td>33,694</td>
<td>Homo sapiens</td>
<td>GemCode</td>
<td>read count</td>
<td></td>
</tr>
</tbody>
</table>

Source: [https://hemberg-lab.github.io/scRNA_seq.datasets/](https://hemberg-lab.github.io/scRNA_seq.datasets/). TPM represents Transcripts Per Million in the raw data type column.

We performed all benchmark analyses on a high-performance computing cluster, of which the computing resources are strictly managed by Univa Grid Engine. The cluster nodes have CPUs of Intel Xeon E5-2680 v3 or Intel Xeon E7-8880 v3. The memory sizes are either 128GB, 256GB, or 1TB. When scheduling analytical jobs for benchmarking, we make sure that the number of cores and allocated memory are enough for the program.

4.3.9.1 Clustering

The clustering accuracy and stability of MIRAC were benchmarked together with several other clustering methods on experimental scRNA-seq datasets listed in Table 4.1.
The following clustering methods are directly applied on the original data without preprocessing.

- MIRAC on 2D t-SNE projection.
- Single-cell consensus clustering (SC3) version 1.7.7 (Kiselev et al. 2017). SC3 is selected for comparison because it was extensively compared with other methods on different experimental datasets.
- K-means clustering on 2D t-SNE projection.
- Hierarchical agglomerative clustering on 2D t-SNE projection.
- Density-based spatial clustering of applications with noise (DBSCAN) (Ester et al. 1996) on 2D t-SNE projection.

Although MIRAC could be directly applied on the expression matrix, dimensionality reduction is able to improve the performance of similarity and density based clustering methods when the number of features is high (Aggarwal, Hinneburg, and Keim 2001). The mathematical influences of the high number of features are briefly described in the previous sections.

Among many dimensionality reduction methods, we chose t-SNE for demonstration in this report in order to visualize the clustering results more clearly. Although t-SNE projections are stochastic and influenced by the perplexity parameter, t-SNE has been extensively used as dimensionality reduction method for scRNA-seq data (Macosko et al. 2015; Cao et al. 2017). Scedar also supports the use of PCA and UMAP for MIRAC clustering, which can be applied just as easily as the t-SNE method.

When benchmarking for accuracy, we cluster the cells in each experimental dataset using the compared clustering methods with a grid of parameters. The maximum
similarities between the clustering results and the cell types from the publications are used to compare the accuracy of different clustering methods. Although taking the maximum increases the chance of overfitting, it resembles the procedure of clustering analysis in practice.

Similarly, when benchmarking for stability, we cluster the cells in each experimental dataset using the compared methods with the same parameters but ten different random states. The similarity of clustering results between different random states are used to compare the stability of different clustering methods.

4.3.9.2 Cluster similarity metrics

We use two cluster similarity metrics, cluster consistent ratio (CCR) and adjusted Rand index (ARI) (Hubert and Arabie 1985) for measuring the accuracy and stability of clustering methods respectively. When we have a coarse reference partition $P_r$ and a finer clustering partition $P_c$, the CCR is computed as the ratio of pairs within each cluster of $P_c$ that are also in the same cluster of $P_r$, with the number of clusters kept the same across compared methods. The ARI is computed with the Python package scikit-learn (Pedregosa et al. 2011) using the mathematical formula given by Hubert and Arabie (Hubert and Arabie 1985).

The reference partitions $P_r$ of real datasets are obtained from their original publications (Table 4.2). The clusters in Deng et al. dataset (Deng et al. 2014) is experimentally determined by manual single cell isolation. The clusters in Pollen et al. (Pollen et al. 2014) is experimentally determined by the cell line or tissue type of the isolated single cells. The clusters in Kolodziejczyk et al. (Kolodziejczyk et al. 2015) are determined by the embryonic stem cell culturing conditions and batches. The clusters in Zeisel et al.
(Zeisel et al. 2015) are determined by the BackSPIN clustering method (Zeisel et al. 2015) and further inspected with domain specific knowledge.

**Table 4.2. Number of clusters for benchmark**

<table>
<thead>
<tr>
<th>Publication</th>
<th># cells</th>
<th># published clusters</th>
<th># cells per cluster</th>
<th># benchmarked clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deng <em>et al.</em> (2014)</td>
<td>268</td>
<td>6</td>
<td>133, 50, 37, 22, 14, 12</td>
<td>6, 8, 9, 10, 11</td>
</tr>
<tr>
<td>Pollen <em>et al.</em> (2014)</td>
<td>301</td>
<td>11</td>
<td>54, 42, 40, 37, 26, 24, 22, 17, 15</td>
<td>11, 12, 13, 14, 15</td>
</tr>
<tr>
<td>Kolodziejczyk <em>et al.</em> (2015)</td>
<td>704</td>
<td>9</td>
<td>93, 90, 82, 82, 81, 79, 72, 66, 59</td>
<td>9, 10, 11, 12, 14</td>
</tr>
<tr>
<td>Zeisel <em>et al.</em> (2015)</td>
<td>3005</td>
<td>9</td>
<td>948, 820, 390, 290, 198, 175, 98, 60, 26</td>
<td>15, 17, 20, 24, 30</td>
</tr>
</tbody>
</table>

We choose CCR to measure clustering accuracy rather than ARI, because ARI greatly penalizes the split of a large cluster in $P_r$ into multiple smaller ones in $P_c$. This behavior of ARI could prevent the evaluation of transcriptomic variabilities within a large group of cells in clustering analysis, which is an important goal of scRNA-seq experiments that cannot be easily achieved by bulk RNA-seq experiments. In addition, we also provide a method in scedar to easily merge multiple clusters together, in case the users found the sub-types of a cell type are very similar to each other.

However, we used ARI to measure clustering stability rather than CCR, because the differences between $P_r$ and $P_c$ are completely caused by different random states, hence splitting a cluster in $P_r$ should be penalized.
MDL is not used as a cluster similarity metric, even though MDL is used in MIRAC to guide the process of finding good partitions. MDL is only used to guide and regularize the merging process of local adjacent sub-clusters by evaluating their similarity, but it is not used as an objective to be optimized globally. Using MDL as a benchmarking metric would introduce a bias as among the various methods, MIRAC is the only method using MDL. Moreover, interpreting relative global MDL differences is not straightforward as it is not only affected by cluster labels, but also transcription levels.

4.3.9.3 Detection of rare transcriptomic profiles

We visualize real datasets before and after removing rare transcriptomic profiles in t-SNE projection and pairwise distance heatmaps, without quantitative evaluations like receiver operating characteristic (ROC) curve, since rare transcriptomic profiles are not well defined with a large number of genes (Aggarwal, Hinneburg, and Keim 2001). Approaches to identify rare data points in low-dimensional spaces (≤ 3) do not scale well to higher dimensions due to the exponentially decreased density of data points in the space and increased instability of distances, which is elaborated in the previous section about mathematical theories on high-dimensional data analysis.

It is important to note that rare transcriptomic profiles are detected to facilitate detailed inspection rather than the removal of them from the data. The visualizations are only used to illustrate the capability of the KNN method for detecting rare transcriptomic profiles.

4.3.9.4 Gene dropout imputation

We simulate gene dropouts using Splatter (Zappia, Phipson, and Oshlack 2017) to obtain a dropout rate around 5%. Then, we benchmark the performance of imputing
gene dropout as two parts, detection and smoothing. On simulated data, because the true gene dropouts are known, we use ROC curve and mean squared errors (MSEs) to characterize the performance of gene dropout detection and smoothing respectively. On real data, we visualize the cells in 2D t-SNE space before and after imputation. The compared methods are KNN gene dropout imputation (KNNGDI) version 0.1.5, SAVER version 1.0.0 (Huang et al. 2018), MAGIC version 1.1.0 (van Dijk et al. 2018), and scImpute version 0.0.6 (W. V. Li and Li 2017).

4.3.10 Data availability

In scedar benchmark, the real scRNA-seq datasets can be obtained from https://hemberg-lab.github.io/scRNA.seq.datasets/ and https://support.10xgenomics.com/single-cell-gene-expression/datasets. The simulated datasets can be generated using scripts at https://github.com/logstar/scedar/tree/master/docs/r_scripts.

4.4 Results

4.4.1 Basic workflow of scedar

We illustrate the basic workflow of using scedar for scRNA-seq exploratory data analysis with the dataset published by Zeisel et al. (Zeisel et al. 2015) (Figure 4.2). The dataset contains the RNA unique molecule identifier (UMI) counts of 19,972 genes in 3005 cells from mouse brain. We selected this dataset for demonstration because it could be clearly visualized in small graphs, although our package is capable of analyzing much larger datasets (Figure 4.3 and Figure 4.4).
In Figure 4.2, each box represents a data analysis step, and they are consecutively executed according to the arrow. The purpose and runtime are listed in the upper ribbon. The code that are essential to the step is listed in the box, and their results are also shown.

The preparation step imports required packages and loads the data. The class SampleDistanceMatrix is one of the core data structures in the package that is used to store the read counts and pairwise distances of an scRNA-seq dataset. Because the pairwise distance computation is delayed until necessary, i.e. lazily loaded, this step only takes 12 seconds. We use cosine distance rather than correlation distance to greatly speed up the computation, since we implemented the computation procedure of pairwise cosine distances completely with numpy linear algebra operations with OpenBLAS backend (Xianyi, Qian, and Yunquan 2012; Q. Wang et al. 2013).

The t-Distributed Stochastic Neighbor Embedding (t-SNE) scatter plot and KNN graph are used to explore the dataset. The cell type labels published by Zeisel et al. (Zeisel et al. 2015) are truncated to fit in the space. We also provide methods to visualize arbitrary statistics, e.g. number of expressed genes, of individual cells as color gradient. The layouts of cells in t-SNE and KNN graph are similar to each other. Although KNN graph is faster than t-SNE, the runtime for t-SNE could be greatly reduced by optimizing its parameter or computation procedure (Linderman et al. 2017).

The MIRAC step clusters the cells and visualizes them with t-SNE scatter plot and pairwise distance matrix heatmap. The heatmap generation procedure in scedar is optimized for large-scale datasets, which is able to generate a heatmap with tens of thousands of columns and rows in a few minutes. Users could also generate heatmaps
for the read count matrix to directly inspect the sparsity of datasets (Figure 4.3B and Figure 4.4B).

The last step identifies cluster separating genes with XGBoost (T. Chen and Guestrin 2016). Users could choose an arbitrary set of clusters to compare, and the genes are ranked by their importance in separating the clusters. Then, the read counts of a gene across clustered labels could easily be visualized by t-SNE scatter plot.

**Figure 4.2.** Demo of scedar. Workflow of using scedar to analyze an scRNA-seq dataset with 3005 mouse brain cells and 19,972 genes generated using the STRT-Seq UMI protocol by Zeisel *et al.* (Zeisel *et al*. 2015). Procedures and parameters that are not directly related to data analysis are omitted. The full version of the demo is available at https://github.com/logstar/scedar/tree/master/docs/notebooks.
Figure 4.3. Scedar analysis of the scRNA-seq dataset containing 44,808 mouse retina cells generated by Drop-seq platform published by Macosko et al. (Macosko et al. 2015). 

(A) t-SNE scatter plot with cell type labels. (B) Read count matrix heatmap with rows as cells, columns as genes, and black color as ≥ 1 reads. (C) t-SNE scatter plot with MIRAC labels. (D) Pairwise cosine distance heatmap with left strip as MIRAC labels and upper strip as cell type labels. (E) t-SNE scatter plot after KNN gene dropout imputation with cell type labels. (F) pairwise cosine distance heatmap with left strip as MIRAC labels and
upper strip as common or rare transcriptomic profile labels. (G) pairwise cosine distance heatmap with rare transcriptomic profiles removed.

Figure 4.4. Scedar analysis of the scRNA-seq dataset containing 68,579 human peripheral blood mononuclear cells generated by 10x genomics GemCode platform published by Zheng et al. (Zheng et al. 2017). (A) t-SNE scatter plot with cell type labels. (B) Read count matrix heatmap with rows as cells, columns as genes, and black color as
≥ 1 reads. (C) t-SNE scatter plot with MIRAC labels. (D) Pairwise cosine distance heatmap with left strip as MIRAC labels and upper strip as cell type labels. (E) t-SNE scatter plot after KNN gene dropout imputation with cell type labels. (F) Pairwise cosine distance heatmap with left strip as MIRAC labels and upper strip as common or rare transcriptomic profile labels. (G) Pairwise cosine distance heatmap with rare transcriptomic profiles removed.

4.4.2 Performance of MIRAC clustering

We benchmarked several clustering methods on the datasets listed in Table 4.1 (Figure 4.5). Each dataset is clustered multiple times with each clustering method to obtain different numbers of clusters (Table 4.2).

The t-SNE based clustering methods are faster than SC3 (Figure 4.5A). Also, the t-SNE based clustering methods have similar runtimes (Figure 4.5A), since the time limiting step is the computation of t-SNE projection.

The cluster consistent ratios (CCRs) of t-SNE based clustering methods are comparable to SC3. The CCRs of Zheng et al. (Zheng et al. 2017) and Macosko et al. (Macosko et al. 2015) datasets are not included in Figure 4.5B, since the SC3 cluster labels are NAs in all 10 runs with different number of clusters. However, the representative MIRAC clustering results of Zheng et al. (Zheng et al. 2017) and Macosko et al. (Macosko et al. 2015) datasets are visualized with t-SNE scatter plot and pairwise distance matrix heatmap (Figure 4.3C and Figure 4.4C). For smaller datasets, the representative MIRAC results are shown in Figure 4.6. Although t-SNE projections obtained with
different random states are distinct from each other, the consistency of clustering results is comparable to SC3 (Figure 4.7).

**Figure 4.5.** Clustering method benchmarks on experimental datasets. (A) Runtimes. (B) CCRs on different datasets, with different points of each dataset representing different numbers of clusters. For each dataset, the numbers of clusters are the same across all compared clustering methods.
Figure 4.6. MIRAC and KNN rare transcriptomic profile detection results of experimental datasets. The sub-figures A, B, and C represent the results of scRNA-seq datasets published by Pollen et al. (Pollen et al. 2014), Deng et al. (Deng et al. 2014), and Kolodziejczyk et al. (Kolodziejczyk et al. 2015) respectively. Within each sub-figure, the plots are 1) t-SNE scatter plot with cell type labels, 2) t-SNE scatter plot with MIRAC
cluster labels, 3) pairwise cosine distance heatmap with left strip as MIRAC labels and upper strip as cell type labels, 4) t-SNE scatter plot with common or rare rare transcriptomic profile labels, 5) pairwise cosine distance heatmap with left strip as MIRAC labels and upper strip as common or rare rare transcriptomic profile labels, 6) pairwise cosine distance heatmap with rare transcriptomic profiles removed.

**Figure 4.7.** Stability of clustering methods on experimental dataset. Similarity between clustering results generated with different random states but the same parameters, quantified by adjusted rand index (ARI) (Hubert and Arabie 1985).

### 4.4.3 Performance of imputing gene dropouts

We benchmarked several gene dropout imputation methods on the simulated 10x Genomics (Error! Reference source not found.) and Drop-seq (Figure 4.9) datasets. K-
nearest neighbor gene dropout imputation (KNNGDI) is faster than other compared methods (Error! Reference source not found.A).

The performance of KNNGDI on detecting gene dropouts is comparable to SAVER and better than scImpute and MAGIC. The ROC curve of scImpute sharply turns around TPR = 0.25 (Figure 4.8B and Figure 4.8B) because its threshold parameters, determining whether a zero entry is a dropout or not, are not sensitive enough to achieve any higher TPRs. Although the AUCs of KNNGDI and SAVER are higher than scImpute and MAGIC, they all have comparable performances when FPRs are lower than 0.05 (Figure 4.8B), except that MAGIC has worse performance on the simulated Drop-seq datasets that have higher sparsity than the Zeisel et al. (Zeisel et al. 2015) dataset (Figure 4.9B).

The MSEs of KNNGDI on correcting gene dropouts are comparable to SAVER and smaller than scImpute and MAGIC (Figure 4.9C and Figure 4.9D). However, these methods all have MSEs many times higher than the MSEs of the observed counts to the true counts, which implies that these methods all introduced many times more reads than the true dropout reads.

None of the compared gene dropout imputation methods greatly improved the separation of different cell types (Figure 4.8C, Figure 4.3E, and Figure 4.4E) in the t-SNE projection of the large, shallow and high dropout rate experimental datasets. On the other hand, the small datasets with higher depth and relatively lower dropout rate do not need gene dropout imputations for cell type separation (Figure 4.6).
Figure 4.8. Gene dropout imputation method benchmarks. (A) Runtimes on 40 simulated 10x Genomics datasets. (B) ROC curves (± standard deviation) of dropout detection on the simulated 10x Genomics datasets. (C) t-SNE scatter plots of the Zeisel et al. (Zeisel et al. 2015) dataset after gene dropout imputations.
Figure 4.9. Benchmark results of gene dropout imputation methods. (A) Runtimes on 40 simulated Drop-seq datasets. (B) ROC curves (± standard deviation) of dropout detection on simulated Drop-seq datasets. (C) and (D) are mean squared error (MSE) ratios of different methods on simulated Drop-seq and 10x Genomics datasets respectively, where the MSE ratio is computed as the MSE of corrected read counts / MSE of true read counts.

4.4.4 Performance of detecting rare transcriptomic profiles

We detected rare transcriptomic profiles in datasets listed in Table 4.1 with the KNN method (Figure 4.3, Figure 4.4, Figure 4.6, and Figure 4.10). Although the detection method has many limitations, rare transcriptomic profiles tend to be points on the t-SNE
scatter plots either far away from the majority of their same types or along the edges of a group of agglomerated points. On the pairwise distance matrix heatmap, rare transcriptomic profiles tend to be small chunks that are distinct from their neighbors, and the heatmap becomes smoother after removing the rare transcriptomic profiles. The detected rare transcriptomic profiles could be further inspected as potential rare cell types and states by comparing with their nearest neighbors.

Figure 4.10. KNN rare transcriptomic profile detection on the Zeisel et al. (Zeisel et al. 2015) dataset. (A) t-SNE scatter plot with colors labeling cell types and markers labeling common or rare transcriptomic profiles. 9.3% cells are marked as rare. (B) Pairwise cosine distance heatmap with left strip as MIRAC labels and upper strip as common or rare transcriptomic profiles labels. (C) Pairwise cosine distance heatmap with rare transcriptomic profiles removed.

4.4.5 Identification of cluster separating genes

We used scedar to identify the genes distinguishing the MIRAC cluster 1, 15, and 22 of the Zeisel et al. (Zeisel et al. 2015) dataset (Figure 4.11 and Table 4.2). In the original
publication, the upper to lower MIRAC clusters 22, 1, and 15 in the t-SNE scatter plot are assigned to microglia, endothelial cells, and astrocytes respectively (Figure 4.8C). We choose these three clusters to inspect one of the discrepancies between cell types and MIRAC clustering results, where the small isolated upper part of the MIRAC cluster 15 is assigned to microglia instead of endothelial cells in the original publication.

The smaller upper isolated part of MIRAC cluster 15 might be a distinct cell sub-type of microglia. Although it expresses a microglia marker gene C1qb (Figure 4.11B) (Beutner et al. 2013), it does not express Mrc1 or Apoe in the same pattern as the MIRAC cluster 22 (Figure 4.11B and Figure 4.11C). According to the transcription levels of the cluster separating genes (Figure 4.11C), the smaller upper isolated part of MIRAC cluster 15, which is located at the top of the cluster 15 rows, has some genes expressed in the same pattern as the microglia, but it also has some other genes expressed distinctly from the microglia.
**Figure 4.11.** Identified genes separating the MIRAC clusters 1, 15, and 22 of the *Zeisel* et al. (Zeisel et al. 2015) dataset. (A) t-SNE scatter plot with color as MIRAC cluster labels and marker shape as compared or not compared. (B) t-SNE scatter plots of the compared clusters with color as $\log_2$(read count + 1) of the corresponding gene and marker shape as MIRAC clusters. (C) Transcription level heatmap of the top 100 important cluster separating genes in the compared cells, with rows as cells ordered by cluster labels and columns as genes ordered by importance. The color gradient is $\log_2$(clip(read count, 1, 100)), where the clip(read count, 1, 100) function changes any
read count below 1 to 1 and above 100 to 100, in order to better compare genes at
different transcription levels.

4.5 Discussion

Comprehensive profiling of the transcriptomes of individual cells within an organism or
tissue by scRNA-seq is able to facilitate the systematic study of physiological or
pathological states (Villani et al. 2017; Baslan and Hicks 2017; Cao et al. 2017).
Previous scRNA-seq experiments identified novel cell types or states (Villani et al.
2017), obtained insights into the regulation of differentiation process (Cao et al. 2017;
Rizvi et al. 2017; Cao et al. 2019; Schiebinger et al. 2019), and inferred molecular
mechanisms of tissue functions (Zeisel et al. 2015; Lake et al. 2018; Haber et al. 2017).

Scedar is able to facilitate gene dropout imputation, rare transcriptomic profile detection,
clustering, and identification of cluster separating genes for scRNA-seq data by
exploiting scalable system design patterns and high-performance computing
architectures. We parallelized time-consuming computational procedures by multi-
processing without excessive copies of shared data in memory. We also decompose the
whole analytical procedure into multiple steps, so that certain steps could be specifically
optimized without repeating others. In addition, intermediate results, such as pairwise
distances and t-SNE projections, are lazy loaded and cached in a unified data structure
to speed up analytical routines, prevent repeated computations, and alleviate the burden
on users to keep track of all intermediate results.

Comparing to other computational tools that were developed or updated for large scale
scRNA-seq data analysis, like PAGODA2 (Fan et al. 2016), Seurat v2.0 (Butler et al.
167

2018), and SCANPY (Wolf, Angerer, and Theis 2018), scedar distinguishes itself with an
additional research focus on developing new analytical methods based on machine
learning. In scedar, we adapted KNN, a typical machine learning algorithm, to impute
gene dropouts and detect rare transcriptomic profiles. MDL principle, an important
concept in computational learning, is applied to cluster single cells (Hansen and Yu
2001). A scalable and sparsity-aware gradient boosted tree system XGBoost (T. Chen
and Guestrin 2016), which implements a typical machine learning algorithm, is used to
identify genes that are able to distinguish different clusters. In addition, these machine
learning methods are able to exploit modern high-performance computing architecture,
which improves the scalability of the package.

In scedar, we developed a clustering algorithm, MIRAC, for scRNA-seq data. MIRAC
clusters observations in three steps: 1) build a tree by hierarchical clustering, 2) divide
the tree into subtrees, and 3) merge similar sub-trees into individual clusters. This
clustering strategy adapts the basic ideas of BIRCH (T. Zhang, Ramakrishnan, and
Livny 1996) and BackSPIN (Zeisel et al. 2015). Instead of building a balanced tree
structure, like the clustering feature tree in BIRCH for further partitioning, MIRAC divides
the tree structure built by hierarchical clustering optionally in a balanced manner (Figure
4.1 in the method section), which simplifies the clustering procedure. In BackSPIN, a
sorted pairwise correlation matrix is recursively bi-partitioned into clusters according to a
criterion based on the normalized sums of correlation coefficients. In contrast, MIRAC
iteratively merges the relatively small sub-clusters according to a criterion based on the
MDL principle, which increases the robustness for determining whether two groups of
observations should be put in the same cluster or not. Especially, when the number of
observations is large within a group, finding an optimal bi-partition is not straightforward,
since there may be multiple distinct sub-groups. Although the performance of MIRAC
under certain metrics is comparable to other clustering algorithms, it is able to provide
distinct clusters that are sensitive to local structures, which could be used as alternative
perspectives to interpret the source of heterogeneity within the dataset.

There are still many possible improvements on scedar. To improve the scalability of
MIRAC, we could provide more efficient methods to obtain the optimal leaf ordering
using linear embedding techniques (Aydin, Bateni, and Mirrokni 2016). To improve the
scalability of the backend data structure, we could extend it with distributed analytic
systems such as Apache Spark (Zaharia et al. 2016). To improve the scalability of KNN
methods, we could also provide implementations with approximate nearest neighbor
methods (Andoni and Indyk 2008; Kushilevitz, Ostrovsky, and Rabani 2000). To
visualize the differences between different clusters, we could plot the fold changes of
gene transcription levels on the pathway maps in the KEGG (Kyoto Encyclopedia of
Genes and Genomes) database (Kanehisa et al. 2016).
5 Systematic analysis of cellular metabolism using single-cell RNA-seq

Abstract: Cellular metabolism encompasses the biochemical reactions and transportation of various metabolites in cells and their surroundings, which are integrated at all levels of cellular functions. We developed a method to systematically simulate cellular metabolism using single-cell RNA-seq (scRNA-seq) data through constraint-based context specific metabolic modeling. We simulated the NAD$^+$ biosynthesis activity in 7 different mouse tissues, and the simulated NAD$^+$ biosynthesis flux levels showed significant linear correlation with experimental measurements in previous research. We also defined metabolic states in mouse brain non-myeloid cell types using dimensionality reduction methods, and we interpreted the metabolic states using the simulated fluxes in multiple metabolic pathways. We further showed that the simulated NAD$^+$ biosynthesis fluxes are reproducible using three different mouse brain non-myeloid scRNA-seq datasets.

5.1 Table of Contents

- Introduction
- Methods
- Results
- Discussion
5.2 Introduction

Cellular metabolism is a critical and integrated component in all biological processes. For example, the Warburg effect is observed in multiple types of tumors (Potter, Newport, and Morten 2016; Warburg, Wind, and Negelein 1927), where the cancer cells have high levels of glucose consumption and lactate secretion. The substrates for various epigenetic modifications are generated by a series of metabolic reactions (Reid, Dai, and Locasale 2017). There are also several lines of evidence showing that psychiatric disorders may have a basis in metabolic dysregulation (Y. Kim et al. 2019; S.-Y. Kim et al. 2017; Clay, Sillivan, and Konradi 2011; Park and Park 2012; Du et al. 2018) and appear as comorbidities with bioenergetic disorders (Postolache et al. 2019; Hung et al. 2014; Wallace 2017; Watson et al. 2019; Hackinger et al. 2018; Hebebrand et al. 2018). Inborn errors of metabolism comprise the largest category of inheritable human diseases, which include over 500 diseases caused by mutations in genes that have functions in metabolism (Childs, Valle, and Jimenez-Sanchez 2001; DeBerardinis and Thompson 2012) that often lead to devastating and complex development comorbidities, including neurodegeneration and physical incapacitation (Pierre 2013).

High-throughput measurement of metabolite concentrations is enabled by mass spectrometry (MS) and nuclear magnetic resonance (NMR) spectroscopy (Wishart 2008; Dettmer, Aronov, and Hammock 2007), whereas the fluxes of metabolic reactions can only be measured in a low-throughput manner through isotope tracing (Jang, Chen, and Rabinowitz 2018; Dai and Locasale 2017). The flux of a metabolic reaction is the rate of metabolite conversion or transportation defined in the reaction, and the fluxes metabolite concentrations may not be correlated (Jang, Chen, and Rabinowitz 2018; Dettmer, Aronov, and Hammock 2007). Metabolic fluxes are experimentally measured by stable
isotope tracing experiments. After introducing isotope-labeled substrates to metabolic systems, the fractions of isotope-labeled intermediate metabolites are measured over time or at steady state by MS or NMR spectroscopy, and the isotope-labeling fractions are further analyzed to infer the metabolic fluxes (Dai and Locasale 2017).

Metabolic modeling methods are able to facilitate the experiment design and result interpretation of metabolic profiling (Bouvin et al. 2015). Although the concentrations of a large number of metabolites can be measured at once using MS and NMR spectroscopy, the development of experimental procedures must be accomplished on a sample by sample basis (Jang, Chen, and Rabinowitz 2018), and an experiment need to be carefully designed to consider the interplay between multiple metabolic pathways (Dai and Locasale 2017; Bouvin et al. 2015). The results of metabolic profiling that contain thousands of metabolite concentrations and multiple reaction fluxes need to be interpreted in a comprehensive mathematical framework, in order to evaluate the metabolic phenotypes. Furthermore, the interpretations obtained from certain experimental conditions can also be accurately extrapolated into other experimental conditions through computational simulation with mathematical metabolic models.

We developed a method to use scRNA-seq data to systematically model cell- and tissue-type specific metabolism through constraint based metabolic modeling. We modeled metabolism as a collection of metabolite conversions within a cellular compartment and exchanges between different cellular compartments (Figure 5.1A and Figure 5.1B). The cellular compartments are the membrane separated spaces, such as cytoplasm, nucleus, and mitochondrial matrix. The comprehensive metabolic models of many organisms are curated by other groups as community resources (King et al. 2016). We reviewed these models and selected a subset of metabolic reactions to represent
cell- or tissue-type specific metabolism based on the transcription levels of relevant enzymes and transporters. The selected metabolic reactions comprise a context specific metabolic model, which can be used to simulate cell- or tissue-type specific metabolic fluxes through various computational methods (Bordbar et al. 2014), such as flux balance analysis (FBA) (Orth, Thiele, and Palsson 2010), flux variability analysis (FVA) (Mahadevan and Schilling 2003), and uniform flux sampling (Schellenberger and Palsson 2009). The simulated metabolic fluxes can be interpreted in biological contexts to hypothesize the metabolic variability between different cell- and tissue-types, and the hypotheses can further be tested by experimental profilings of metabolism.
Figure 5.1. Example metabolic model. (A) Network representation of the example metabolic model. There are four compartments in this example, including extracellular space, cytoplasm, mitochondria, and nucleus. Metabolites are transported from one compartment to another by transporters. Enzymes catalyze the conversions between metabolites. Reactions can be either irreversible or reversible. Three types of reactions are mass unbalanced, including exchange, demand and sink reactions. The metabolite demand is capable of consuming cytosolic metabolite D, and the metabolite D demand reaction export cytosolic metabolite D outside of the model. The metabolite A sink is capable of providing or consuming cytosolic metabolite A, and the metabolite A sink
reaction imports cytosolic metabolite A into the model or exports cytosolic metabolite A outside of the model. The metabolite exchange reactions are capable of importing metabolites into the extracellular space from the external environment or exporting metabolites in the extracellular space into the external environment. (see Method section for more details about exchange, demand and sink reactions) (B) Reactions in the metabolic model. The text in the brackets represents the compartment of the metabolite: [ex] means extracellular space; [cyt] means cytoplasm; [nuc] means nucleus; [mito] means mitochondria. The color shading represents the correspondence between the reaction formulae and stoichiometric coefficients of the metabolites. (C) Stoichiometric matrix $S$ of the metabolic model. Each entry represents the stoichiometric coefficient of its corresponding metabolite (row) in its corresponding reaction (column) (see Method section for details).

We applied our metabolic modeling method on several mouse scRNA-seq datasets to simulate the metabolic fluxes of various cell and tissue types. We constructed context specific metabolic models for each cell and tissue type in the scRNA-seq datasets using CORDA (Schultz and Qutub 2016) and comprehensive mouse metabolic model iMM1415 (Sigurdsson et al. 2010). The simulated NAD$^+$ biosynthesis fluxes of 7 different mouse tissues show significant linear correlation with experimental measurements (Mori et al. 2014), in two different approaches of model construction using the Tabula Muris scRNA-seq dataset (Schaum et al. 2018). In one approach, we constructed models using the cells in each tissue type and directly simulated the tissue-type specific metabolic fluxes. In the other approach, we constructed models using the cells in each cell type, and we used these cell-type specific models to simulate cell-type specific
metabolic fluxes. Then, we computed the tissue-specific metabolic fluxes as the average of comprising cell-type specific metabolic fluxes weighted by the number of cells in each cell type. In addition, we also evaluated the variance of the simulated fluxes by bootstrapping the cells in each cell type or tissue, and the flux mean and standard deviation of the bootstrapping samples are able to illustrate the metabolic variations within each group of cells and between groups of cells. We applied the developed metabolic modeling approach to explore the metabolism in mouse brain cells, using *Tabula Muris* and Allen Brain Mouse Atlas scRNA-seq datasets. We defined three metabolic states in mouse brain non-myeloid cell types using the clusters in UMAP (McInnes and Healy 2018) dimensionality reduced metabolic gene read counts, and we interpreted the metabolic states using the simulated fluxes in multiple metabolic pathways. The simulated NAD$^+$ biosynthesis fluxes of 11 mouse brain cell types are reproducible when computed using different scRNA-seq datasets.

5.3 Methods

5.3.1 Context-specific metabolic model construction using scRNA-seq data

Modeling cellular metabolism is motivated by the identification of various metabolic pathways in cells such as glycolysis, tricarboxylic acid (TCA) cycle, and oxidative phosphorylation. In a curated model of human metabolism published in 2018, there are 5,835 metabolites and 10,600 reactions (Brunk et al. 2018). In addition to the large number of metabolites and reactions, the reactions interact in a non-independent and non-additive manner (Le Novère 2015), and the metabolites are extensively consumed in other biological processes, such as nucleotides in transcription, amino acids in
translation, NAD$^+$ in parylation (Gibson and Kraus 2012), and acetyl-CoA in histone acetylation (Reid, Dai, and Locasale 2017). Furthermore, metabolites are constantly exchanged across cellular membranes through transporters and diffusion. Mathematically describing the current knowledge of cellular metabolism is able to facilitate experiment design and interpretation of results.

Mathematical modeling of cellular metabolism is based on the principles of kinetics in physical chemistry. The relationships between metabolite concentrations and reaction fluxes are written as a system of differential equations as the following (Costa, Hartmann, and Vinga 2016),

$$\frac{d[X_i]}{dt} = \sum_j S_{i,j} \cdot v_j([E], [X], \tilde{k}).$$

In the differential equations:
- $[X_i]$ is the concentration of the $i$-th metabolite
- $t$ is time.
- $S$ is the stoichiometric matrix of the metabolic model (Figure 5.1C). Each row represents a metabolite, and each column represents a reaction. The entry $S_{i,j}$ is the stoichiometric coefficient of the $i$-th metabolite in $j$-th reaction. $S_{i,j}$ is positive if the $i$-th metabolite is on the right-hand side of the reaction, otherwise negative.
For example, the 3rd reaction in Figure 5.1B is “A[cyt] + B[cyt] $\rightarrow$ C[cyt]”, so $S_{1,3} = -1$, $S_{2,3} = -1$, and $S_{5,3} = 1$.
- $v_j$ is the reaction flux, or rate, of the $j$-th reaction, converting left-hand-side reactants into right-hand-side products. Fluxes are dependent on the concentrations of metabolites ($[X]$) and enzymes ($[E]$) and the kinetic constants of the enzymes ($\tilde{k}$). The enzyme concentrations can be considered as constants
within a certain period of time or variables dependent on parameters relevant to protein degradation. When the reaction has only one reactant and one product and is catalyzed by one enzyme, the flux of the reaction can be computed with

\[ \nu = [E] \cdot k_{\text{cat}} \cdot \frac{[S]}{[S] + K_m} \]

where \([E]\) and \([S]\) are the concentrations of the enzyme and substrate respectively. \(k_{\text{cat}}\) is turnover number, which is a kinetic constant of the enzyme representing the number of catalytic cycles can be performed by the enzyme within a unit of time. \(K_m\) is Michaelis constant, which is a constant representing the affinity between the enzyme and its substrate.

The differential equation system defines the changes of metabolite concentrations and reaction fluxes over time, given initial concentrations of metabolites and enzymes, stoichiometric coefficients, and kinetic parameters (Saa and Nielsen 2017).

There are mainly two methods to use the differential equation system to model cellular metabolism, either kinetic modeling or constraint-based modeling. In kinetic modeling, experimentally measured \([X]\) and \(\dot{\nu}\) of a cell type are plugged into the differential equation system, and the changes of \([X]\) and \(\dot{\nu}\) are simulated over time to characterize the metabolism of the cell type. In constraint-based modeling, the differential equation system is analyzed under steady metabolite concentration state, in which the changes of metabolite concentrations are all equal to 0. The steady metabolite concentration state is biologically meaningful, because the changes of metabolite concentrations are relatively slower than the changes of metabolic fluxes (Fell 2005; Edwards and Palsson 2000; Palsson and Lightfoot 1984). Under steady state, the differential equation system becomes a system of linear equations with the fluxes as variables, as the following
Although $v_j$ can still be computed by kinetic laws given $([E], [X], \bar{k})$, we can compute $v_j$ without $([E], [X], \bar{k})$ by assuming that cellular metabolism optimizes the fluxes of a set of reactions, such as the reactions involved in cell growth, through linear optimization with additional lower and upper bounds of reaction fluxes. The lower and upper bounds of reaction fluxes are called constraints in a linear programming problem, and the bounds can be experimentally determined or computationally estimated (Hackett et al. 2016).

We selected constraint-based modeling to infer cell-type specific metabolic fluxes using scRNA-seq data, because constraint-based modeling requires less initial information than kinetic modeling. Kinetic modeling requires kinetic constants of enzymes and concentrations of metabolites and enzymes that are not available for mouse or human, whereas constraint-based modeling only requires the stoichiometric coefficients and flux constraints. Although the flux constraints are also not available for different mouse or human cell types, the computed fluxes through constraint-based modeling are robust to inaccurate flux constraints due to the robustness and redundancy of metabolic networks (Edwards and Palsson 2000).

In constraint-based metabolic flux computation, the cell-type specificity is determined by the enzyme transcription levels in different cell types characterized by scRNA-seq. We infer whether an enzyme is expressed in a cell type or not from scRNA-seq data, based on the average read counts of the enzyme in the single cells with the same cell type. Considering that enzymes are transcribed in vivo at distinct levels, we applied CORDA to refine the determination of whether an enzyme is expressed or not in a cell type (Schultz and Qutub 2016), through a linear programming setting to minimize the
inclusion of lowly transcribed enzymes while favoring the inclusion of important enzymes. The enzymes that are determined to be expressed in the cell type formulate a cell-type specific stoichiometric matrix $S$, which is further used for flux computation in various optimization settings. The upper and lower bounds of the fluxes are not customized for the cell type based on the enzyme transcription levels, but such procedures can be integrated in the future to improve the sensitivity of flux computation to distinct transcriptomic profiles.

The constraint-based flux computation is either based on optimization or sampling procedures under steady state. In optimization procedures, flux balance analysis (FBA) is the foundation of various other methods. In FBA, we assume that cellular metabolism under steady state is optimized for certain objectives, such as cell growth and ATP production. Therefore, the metabolic differential equation system can be written as a linear programming problem in Figure 5.2, and the problem can be efficiently solved in polynomial time using free or commercial software to obtain the maximum values of objective fluxes (Meindl and Templ 2012). The maximum objective fluxes are cell-type specific, because different cell types have different stoichiometric matrix $S$. In sampling procedures, metabolic fluxes are sampled from the allowable solution space defined by steady state $S \cdot v = 0$ and flux constraints $l \leq v \leq u$ based on certain statistical distributions (Megchelenbrink, Huynen, and Marchiori 2014).
Figure 5.2. Flux balance analysis. (A) Flux balance analysis procedure, adapted from the illustration by Orth et al. (Orth, Thiele, and Palsson 2010). $Z$ is the objective of the linear programming problem, and it is a linear combination of the fluxes of one or more
metabolic reactions. $S \cdot v = 0$ describes steady state, in which the metabolite concentrations do not change over time. $l \leq v \leq u$ describes the flux constraints, so that the optimized flux values will not be infinite. (B) Example flux balance analysis on the metabolic model in Figure 5.1A. The objective is to maximize the flux of reaction 11, which is the demand reaction of metabolite D. The steady state is described as the equation $S \cdot v = 0$, which can be converted to a system of linear equations. The constraints are the reaction flux bounds, which are assigned according to the directionality of reactions. The optimization results are shown on the network representation of the metabolic model.

5.3.2 Exchange, demand and sink reactions

Constraint based metabolic models are open chemical systems, which are able to import metabolites from external chemical systems and export metabolites to external chemical systems. The interactions between the models and the external chemical systems are mediated through exchange, demand, and sink reactions. Such reactions are written as mass unbalanced reactions in the models (Figure 5.1B), but they are actually balanced with the external chemical systems. The exchange, demand, and sink reactions are also biologically meaningful.

The exchange reactions are reversible reactions with the metabolites in extracellular space as reactants, which represent the availability and consumability of metabolites in the external chemical system. The exchange reactions are defined by the culturing medium or extracellular environment of the modeled cells. The extracellular environment
is distinct from the extracellular space in constraint based models (Figure 5.1A). The metabolites and reactions in extracellular space are internal to the model, representing the interactions between the modeled cells and their extracellular space. The extracellular environment is external to the model. The exchange reactions transport metabolites between extracellular space and extracellular environment, which define the availability of metabolites for the modeled cell (Thiele and Palsson 2010; Sigurdsson et al. 2010). If an exchange reaction carries a positive flux, it means its corresponding reactant is exported from the extracellular space into the extracellular environment. If an exchange reaction carries a negative flux, it means its corresponding reactant is imported from the extracellular environment to the extracellular space.

The demand reactions are irreversible reactions with the metabolites inside the modeled cells as reactants (Thiele and Palsson 2010), which represent the accumulation or consumption of metabolites inside the modeled cells. Conventionally, the direction of an irreversible reaction is from the left hand side of the formula to the right hand side, so irreversible reactions can only carry non-negative fluxes. If a demand reaction carries a positive flux, it means the reactant is accumulated in the external chemical system or consumed by the external chemical system. We optimize demand reactions to simulate the biosynthesis of metabolites.

The sink reactions are reversible reactions with the metabolites inside the modeled cells as reactants (Thiele and Palsson 2010), which have different meanings when carrying positive or negative fluxes. If a sink reaction carries positive fluxes, it has the same meaning as a demand reaction with the same flux. If a sink reaction carries negative flux, it means that the reactant is provided by the external chemical system.
The definitions of exchange, demand and sink reactions may be slightly different in different publications, but their usage is the same, which is to enable interactions between the constraint based metabolic models and their external chemical systems.

5.3.3 Construction of a context specific metabolic model

We construct context specific metabolic models by extracting a subset of reactions from a reference genome scale metabolic model based on the single-cell gene transcription levels. A reference model contains all known metabolic reactions that can occur in an organism, and a context specific model only contains the metabolic reactions in a specific tissue or cell type. The reactions in a context specific model comprise a subset of the reactions in the reference model, and the subset is selected according to the transcription levels of enzymes in a specific tissue or cell type. Therefore, a context specific metabolic model more accurately represents the metabolism of its corresponding cell type or tissue.

Context specific metabolic models can represent the transcriptome of any group of cells, which is not limited to cell type, tissue, or organ. Although the cellular organisational levels are different, the model abstracts an open chemical system that is able to import metabolites from external sinks, convert the metabolites internally for various biological objectives, and export metabolites to external demands (Figure 5.1A). The sinks of metabolites can be interpreted as large pools of metabolites that are outside the chemical system defined by the metabolic model, and the sink reactions are able to import the metabolites from the external pools to the system. The demands of metabolites can be interpreted as external biochemical components that are able to consume the metabolites, e.g. histone deacetylases use NAD⁺ to remove acetyl groups.
from histones (Reid, Dai, and Locasale 2017), and the demand reactions are able to consume the metabolites in the system.

We group the cells in the Tabula Muris dataset (Schaum et al. 2018) by the tissue type and cell ontology class annotations of each cell. We constructed context specific models for tissue types and cell ontology classes. The tissue type models represent the overall metabolism of the cells in the tissue, and the cell ontology class models represent the metabolism of specific cell types. We removed cell ontology classes with less than 35 single cells in the FACS dataset, in order to ensure that the transcriptomic profiles of ontology classes can be accurately inferred from the transcriptomic profiles of individual cells.

We convert the transcriptomic profiles of a group of cells into gene expression confidence scores based on the mean read count rankings of each gene. For each group of cells, we rank the genes by their mean read counts. Then, we assign discrete confidence scores to each gene to represent the likelihood that the gene is expressed in that group of cells. We assign the confidence score of 3 to the top 2000 genes, which represent that those genes are highly likely to be expressed in the cell group. Similarly, we assign the confidence score of 2 to the following 4000 genes, which represent the medium likelihood of expression. We assign the confidence score of 1 to the following 2000 genes, which represent the low likelihood of expression. We assign the confidence score of 0 to the following 1000 genes, which represent the unknown likelihood of expression. Finally, we assign confidence score of -1 to the rest of the genes, which represent that the genes are unlikely to be expressed in the cell group. The choice of the numbers of reactions with different confidence scores depends on the dropout rate and the amount technical variance in the dataset. If the dropout rate and technical variance is
high, it is more likely that important enzymatic genes will be ranked low. These parameters are evaluated for each dataset to balance the sensitivity and robustness of the method.

The gene expression confidence scores are further converted to metabolic reaction confidence scores based on the associations between reactions and genes. The associations are represented as boolean algebraic operations with genes as operands. For example, the gene association of the reaction “ATP + D-Fructose 6-phosphate <=> ADP + D-Fructose 1,6-bisphosphate”, which is catalyzed by phosphohexokinase, is “56421 or 18641 or 18642”, where the numbers are Entrez Gene IDs. When evaluating the boolean algebraic operations, the result of an “or” operation is the maximum gene confidence score of two operand genes, and the result of an “and” operation is the minimum gene confidence score of two operand genes.

We construct context specific metabolic models based on the reaction confidence scores using the Cost Optimization Reaction Dependency Assessment (CORDA) algorithm (Schultz and Qutub 2016). We choose CORDA among many other construction algorithms (Pacheco, Pfau, and Sauter 2015) due to its efficiency and discretization of reaction confidence scores. CORDA uses linear programming to construct context specific metabolic models, so its time complexity is tractable in constructing thousands of models. The discretization of reaction confidence scores greatly improves the robustness of model construction based on scRNA-seq data, by reducing the variances between different cells within a group. To implement the construction process, we use python programming language version 3.6.8, cobrapy version 0.14.1 (Ebrahim et al. 2013), python implementation of CORDA version 0.4.2
(https://github.com/resendislab/corda), and a commercial linear programming solver Gurobi Solver version 8.1.

The CORDA constructed models are further modified for quality and simulation purposes. In order to ensure the validity, if essential metabolites, such as H$_2$O and NAD$^+$, and relevant reactions are absent in the CORDA generated models, we will add them into the models before further simulation. When simulating the biosynthesis fluxes of certain metabolites, such as NAD$^+$ and ATP, we add in demand reactions that consume the corresponding metabolites in the CORDA generated models.

5.3.4 Jaccard dissimilarity between two context specific metabolic models

For each context specific metabolic model, we construct an indicator vector to represent the presence and absence of each reaction in the reference metabolic model. If the i-th reaction is present in the context specific model, the i-th entry of the indicator vector has value 1. If the i-th reaction is absent in the context specific model, the i-th entry of the indicator vector has value 0. The Jaccard dissimilarity between two context specific metabolic models is computed as the Jaccard dissimilarity of their reaction indicator vectors, as the following

$$\frac{c_{10} + c_{01}}{c_{11} + c_{01} + c_{10}},$$

where $c_{ab}$ is the count of occurrences of the i-th entry of the first reaction indicator vector is equal to $a$ and the i-th entry of the second reaction indicator vector is equal to $b$ for $i \leq$ the number of total reactions (Kosub 2019). The models are clustered on the Jaccard dissimilarities by hierarchical clustering with complete linkage, which are plotted in Figure 5.5A and Figure 5.5B as heatmaps and dendrograms.
5.3.5 Computational analysis of metabolic fluxes

We use flux balance analysis (FBA) (Orth, Thiele, and Palsson 2010) (Figure 5.2) to simulate the metabolic fluxes of context specific metabolic models. The FBA objectives are defined to maximize the fluxes of demand reactions of certain metabolites, in order to simulate the maximum amount of metabolites that can be synthesized by the model under steady state. The metabolic variations between different context specific models are represented by the differences in FBA simulated fluxes. To implement the FBA, we use Python programming language version 3.6.8, Cobrapy version 0.14.1 (Ebrahim et al. 2013) and a commercial linear programming solver Gurobi Optimizer version 8.1.

The NAD\(^+\) biosynthesis flux is simulated by adding a NAD\(^+\) demand reaction in the compartment of cytosol and applying FBA to optimize the NAD\(^+\) demand reaction. The simulated fluxes of the NAD\(^+\) demand reaction is compared with the empirically measured NAD\(^+\) biosynthesis fluxes represented by the sum of NADS and NMNAT fluxes (Mori et al. 2014). The linear regression between the simulated NAD\(^+\) fluxes and the empirically measured ones are computed by the linregress function in scipy package version 1.2.1. The p-value is computed using Wald Test with t-distribution, with the null hypothesis that the slope is zero.

5.3.6 Estimate metabolic variation with bootstrapping

In order to estimate the metabolic variation within a group of cells, we construct context specific models based on the bootstrapping samples of the group of cells, and we simulate metabolic fluxes using the constructed models. For each group of cells, we generate 20 bootstrapping samples by sampling 80\% of the cells with replacement, and we construct context specific models based on each sample and simulate the metabolic
fluxes using the procedures described above. Then, we use the standard deviation of the simulated fluxes of the bootstrapping samples to describe the metabolic variation within the group.

5.3.7 Explore metabolic states in brain non-myeloid cells

We define the metabolic states of brain non-myeloid cells by 6 core metabolic pathways in the iMM1415 model, including Citric Acid Cycle, Fatty acid oxidation, Glycolysis/Gluconeogenesis, NAD Metabolism, Oxidative Phosphorylation, and Pentose Phosphate Pathway. We computed the UMAP embeddings of brain non-myeloid cells using all genes or 182 metabolic genes in the selected pathways. We define oligodendrocyte main group as the oligodendrocytes with the first component of UMAP computed using all genes (Figure 5.10C) less than -5, and we define other oligodendrocytes as sub group. We also computed UMAP embeddings of brain non-myeloid cells using 1,000 sets of randomly selected 182 genes, and we characterized the differences between random gene embeddings and metabolic gene embeddings using the adjusted rand index (ARI) between their k-means clustering results with three clusters (Figure 5.11A).

We plotted the read counts of genes in the selected metabolic pathways as a heatmap (Figure 5.9A) using scedar. Each entry in the heatmap represents the transformed read count of its corresponding row of gene and column of cell. We annotated several genes that show noticeable cell type specific expression. The read counts are log₂ transformed and clipped between 0 and 10, i.e. the read counts > 2^{10} will be illustrated as 2^{10}, in order to make the read counts comparable on the graph.
We plotted the FBA simulated fluxes of the reactions in the selected metabolic pathways as a heatmap (Figure 5.9B) using scedar. Each entry represents the mean FBA simulated fluxes of 20 bootstrapped models of its corresponding column of cell type and row of reaction, and the mean FBA fluxes are z-score transformed for each reaction to make different reaction fluxes comparable on the graph. The FBA fluxes are simulated by optimizing each selected metabolic reaction as the objective, and the reversible reactions, of which the two directions usually represent catabolic and anabolic purposes, are optimized for both directions separately. Then, we performed complete linkage hierarchical clustering on the cell types, based on the correlation distances between the mean FBA simulated fluxes of cell types without z-score transformation. Also, we ordered the reactions within each metabolic pathway based on their hierarchical cluster results computed using the same procedure as the cell type clustering.

We computed the linear regression between the mean FBA fluxes and mean read counts in brain non-myeloid cell types for each gene and its corresponding reaction (Figure 5.11B, Figure 5.11C, and Figure 5.11D). Also, we computed the linear regression between mean FBA fluxes and 1000 sets of randomly selected 182 genes (Figure 5.11E), and we plotted the distribution of minimum FDR of each set of random genes to compare with the metabolic genes.

5.3.8 Data availability

We obtained the Tabula Muris mouse single-cell RNA-seq dataset from https://tabula-muris.ds.czbiohub.org (Schaum et al. 2018). The version 8 dataset of the Smart-seq2 sequencing results of the FACS sorted cells is used to construct context specific metabolic models. The sequenced 53,760 single cells are isolated from 20 organs of 7 mice.
The in vivo NAD biosynthesis fluxes in different mouse tissues are obtained from the experimental measurements made by Mori et al (Mori et al. 2014). The fluxes of the two final NAD biosynthetic reactions, which are respectively catalyzed by nicotinamide mononucleotide adenylyltransferase (NMNAT) and NAD synthetase (NADS), are used to compare with our computationally simulated fluxes.

The mouse genome scale metabolic model iMM1415 is obtained from Biochemical, Genetic and Genomic (BiGG) Models database (King et al. 2016). The iMM1415 model is reconstructed by Sigurdsson et al (Sigurdsson et al. 2010), which contains 3,726 reactions and 2,775 metabolites in 8 cellular compartments. In this model, 2,210 reactions are associated with 1,375 genes, and their associations are used to construct context specific metabolic models.

5.4 Results

5.4.1 Tissue and cell type specific metabolic models

We constructed context specific metabolic models for 20 tissue types and 111 cell types using the 44,728 filtered and annotated FACS sorted cells that are sequenced with Smart-Seq2 (Schaum et al. 2018). We characterized the differences between different models by the Jaccard dissimilarities of the metabolic reactions, which are illustrated as heatmaps (Figure 5.3A and Figure 5.3B). We also illustrated the differences between different models by applying UMAP dimensionality reduction on the Jaccard dissimilarities (Figure 5.3C) (McInnes and Healy 2018).
Figure 5.3. Tissue and cell type context-specific metabolic models constructed using Tabula Muris scRNA-seq dataset (Schaum et al. 2018). (A) Pairwise Jaccard dissimilarities between tissue type specific metabolic models. (B) Pairwise Jaccard dissimilarities between cell type specific metabolic models. (C) UMAP 2D embeddings of the tissue and cell type specific metabolic models using the Jaccard dissimilarities between their reaction indicators. The text labels are close to the tissue type specific models, and the lines connect tissue specific metabolic models and their corresponding cell type specific models.

The variations between different metabolic models showed that the metabolic modeling procedure is sensitive to the differences in transcriptomic profiles. The UMAP
embeddings of metabolic models (Figure 5.3C) separated different cells in a similar pattern as the UMAP embeddings computed by the cosine distance of single-cell gene read counts of the metabolic genes in iMM1415 (Figure 5.4B). The UMAP embeddings of metabolic models (Figure 5.3C) and 1,375 metabolic genes (Figure 5.4B) are less capable of separating different tissue types, comparing to the UMAP embeddings of all 44,728 genes, which implies a less diverse space of cellular metabolic states compared to cell types. The mean number of reactions in all cell type models is 1,376, and the standard deviation is 76 (Figure 5.5A). The mean and standard deviation of the number of reactions in all tissue type models are 1,383 and 83 (Figure 5.5A).
Figure 5.4. UMAP 2D embeddings of the transcriptomic profiles of different cells in Tabula Muris dataset. (A) UMAP embeddings computed using all genes. (B) UMAP embeddings computed using only genes in the iMM1415 metabolic model.
In order to deal with the high drop-out rate in scRNA-seq, we agglomerated multiple cells together to construct a metabolic model. The mean number of expressed genes in all cell types is 9,131, and the standard deviation is 1,490 (Figure 5.5B). In all tissue types, the mean number of expressed genes is 10,599, and the standard deviation is 993 (Figure 5.5B). The nearly 10,000 mean number of expressed genes suggests a low gene dropout rate with regard to cell types and tissue types. The median number of cells in all cell types and tissue types is 178 and 1,630 respectively (Figure 5.5C), which suggests that the probability is low for a gene to be stochastically dropped out in all cells of a cell or tissue type.

**Figure 5.5.** General statistics of the Tabula Muris scRNA-seq dataset and metabolic models. (A) Histogram of the number of reactions in cell- and tissue- type specific
metabolic models constructed using the Tabula Muris dataset. (B) Histogram of the number of genes with mean read counts > 3 in different cell and tissue types in Tabula Muris dataset. (C) Histogram of the number of cells in different cell and tissue types in Tabula Muris dataset.

5.4.2 Simulated tissue specific NAD$^+$ biosynthesis fluxes

We simulated the tissue specific NAD$^+$ biosynthesis fluxes show significant ($p < 0.05$) linear correlation with empirically measured ones (Mori et al. 2014) (Figure 5.6A and Figure 5.6B). The R$^2$ value is higher in tissue type model simulated fluxes, which may be caused by the inaccurate estimations of reaction confidence scores in certain cell types due to the low number of single cells in the dataset. When the number of cells is low, the estimation of gene expression confidence scores is strongly affected by the noise in scRNA-seq data. Also, the number of cells in each cell type of a tissue may not accurately represent the in vivo composition (Newman et al. 2019), which would result in the inaccurate weights in the computation of tissue specific fluxes using cell type model simulated fluxes.

The linear correlations between experimentally measured and FBA simulated NAD$^+$ biosynthesis fluxes cannot be directly explained by the expression levels of individual genes. The linear correlations are not significant between empirically measured NAD$^+$ biosynthesis fluxes and the expression levels of the enzymes that are catalyzing the last step of the NAD$^+$ biosynthesis (Figure 5.7), including Nmnat1, Nmnat2, Nmnat3, and Nadsyn1 (Table 5.1). Although the expression levels of Qprt and Nmrk1 show significant
linear correlations to the empirically measured NAD\(^+\) biosynthesis fluxes, the false
discovery rates (FDRs) are not significant when Qprt and Nmrk1 are tested together with
the key enzymes in NAD\(^+\) biosynthesis (Table 5.1 and Figure 5.7) (Mori et al. 2014), all
genes, or metabolic model iMM1415 genes. The lowest FDRs in the linear correlations
between all 22,454 genes and empirically measured NAD\(^+\) biosynthesis fluxes is
0.071542. Although there are 72 genes in the iMM1415 with FDR < 0.05, interpreting
these significant linear correlations is not straightforward, and the key enzymes in NAD\(^+\)
biosynthesis all have FDRs > 0.09.

Figure 5.6. FBA simulated NAD\(^+\) biosynthesis fluxes. (A) Linear correlation between
tissue type model simulated NAD\(^+\) biosynthesis fluxes and empirically measured ones.
(B) Linear correlation between cell type model simulated NAD\(^+\) biosynthesis fluxes and
empirically measured ones, in which the simulated tissue fluxes are the weighted mean
of cell type fluxes by the number of cells in each tissue.
Figure 5.7. NAD biosynthesis pathway in mouse. The involved genes are Nmrk1 (nicotinamide riboside kinase 1), Nmrk2 (nicotinamide riboside kinase 2), Qprt (quinolinate phosphoribosyltransferase), Naprt1 (nicotinate phosphoribosyltransferase), Nmnat1 (nicotinamide nucleotide adenylyltransferase 1), Nmnat2 (nicotinamide nucleotide adenylyltransferase 2), Nmnat3 (nicotinamide nucleotide adenylyltransferase 3), Nadsyn1 (NAD synthetase 1), and Nampt (nicotinamide phosphoribosyltransferase).
Table 5.1. Linear correlations between empirically measured tissue specific NAD\(^+\) biosynthesis fluxes and \(log_2\) mean read counts + 1 of key enzymes in NAD\(^+\) biosynthesis pathway.

<table>
<thead>
<tr>
<th>Gene symbol</th>
<th>R</th>
<th>(R^2)</th>
<th>p-value</th>
<th>FDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qprt</td>
<td>0.885</td>
<td>0.783</td>
<td>0.008</td>
<td>0.068</td>
</tr>
<tr>
<td>Nmrk1</td>
<td>0.851</td>
<td>0.724</td>
<td>0.015</td>
<td>0.068</td>
</tr>
<tr>
<td>Naprt1</td>
<td>0.659</td>
<td>0.434</td>
<td>0.107</td>
<td>0.322</td>
</tr>
<tr>
<td>Nmnat1</td>
<td>0.596</td>
<td>0.355</td>
<td>0.158</td>
<td>0.356</td>
</tr>
<tr>
<td>Nmnat2</td>
<td>-0.279</td>
<td>0.078</td>
<td>0.545</td>
<td>0.645</td>
</tr>
<tr>
<td>Nampt</td>
<td>0.310</td>
<td>0.096</td>
<td>0.499</td>
<td>0.645</td>
</tr>
<tr>
<td>Nadsyn1</td>
<td>0.380</td>
<td>0.144</td>
<td>0.401</td>
<td>0.645</td>
</tr>
<tr>
<td>Nmrk2</td>
<td>-0.260</td>
<td>0.068</td>
<td>0.573</td>
<td>0.645</td>
</tr>
<tr>
<td>Nmnat3</td>
<td>0.109</td>
<td>0.012</td>
<td>0.816</td>
<td>0.816</td>
</tr>
</tbody>
</table>

5.4.3 Metabolic variations within tissue and cell types estimated by bootstrapping

We estimated the metabolic variations within different tissue and cell types by bootstrapping. The standard deviations of bootstrapping sample fluxes are used to illustrate the extent of metabolic variations, and the means are used to compute the significance of linear correlation with empirically measured fluxes (Figure 5.8A and Figure 5.8B). The linear correlation significance varies in each bootstrapping run (Figure 5.9B and Figure 5.9C), which may be caused by the signal-to-noise ratio in the
bootstrapping samples. In addition, the bootstrapping fluxes provide the estimates of the metabolic variations (Figure 5.8C and Figure 5.8D), which is informative in exploring the heterogeneous scRNA-seq datasets.

The distribution of NAD$^+$ biosynthesis fluxes in bootstrapping samples implies that a tissue type may be composed of cells in multiple metabolic states (Figure 5.8C and Figure 5.8D). The fluxes are separated into high and low groups in several tissues, such as kidney and skeletal muscle. Similar separations are also shown in the fluxes of bootstrapping samples of several cell types (Figure 5.9A), such as kidney macrophage and lung endothelial cell. Therefore, we looked further into the metabolic states, defined by gene expression levels and simulated metabolic fluxes, of the comprising single cells of tissue types and cell types.
Figure 5.8. FBA simulated NAD⁺ biosynthesis fluxes with bootstrapping. (A) Linear correlation between tissue type model simulated NAD⁺ biosynthesis fluxes and empirically measured ones. The error bars are the standard deviation of the bootstrapping samples. (B) Linear correlation between cell type model simulated NAD⁺ biosynthesis fluxes and empirically measured ones, in which the simulated tissue fluxes are the weighted mean of cell type fluxes by the number of cells in each tissue. The error bars are the standard deviation of the bootstrapping samples. (C) Distribution of the tissue type model simulated NAD⁺ biosynthesis fluxes in all bootstrapping samples. (D) Distribution of the cell type model simulated NAD⁺ biosynthesis fluxes in all bootstrapping samples, in which the simulated tissue fluxes are the weighted mean of cell type fluxes by the number of cells in each tissue.
**Figure 5.9.** Bootstrapping results of metabolic modeling. (A) Distribution of the cell type model simulated NAD⁺ biosynthesis fluxes in all bootstrapping samples. (B, C) Distribution of the p-values (B) and R² (C) of linear correlation between empirically measured NAD⁺ fluxes and 15 bootstrapping runs of tissue type models. (D, E) Distribution of the p-values (B) and R² (C) of linear correlation between empirically measured NAD⁺ fluxes and 15 bootstrapping runs of cell type models.

### 5.4.4 Metabolic states within brain non-myeloid cell types

Three metabolic states of the brain non-myeloid cells are defined by metabolic genes (**Figure 5.10C** and **Figure 5.10D**). The UMAP embeddings computed using 182 selected metabolic genes separate 3,401 brain non-myeloid cells into three clusters, which are significantly different from the clusters computed by randomly selected sets of 182 genes (**Figure 5.12A**). The selected 182 metabolic genes are associated with the reactions in 6 core metabolic pathways of iMM1415, including citric acid cycle, fatty acid oxidation, glycolysis/gluconeogenesis, NAD metabolism, oxidative phosphorylation, and pentose phosphate pathway. The cell types, except for oligodendrocytes and endothelial cells, are less separated than the UMAP embeddings computed using all genes (**Figure 5.11A** and **Figure 5.11B**), which implies that different cells may share similar metabolic states. In the UMAP embeddings computed using all genes (**Figure 5.11B**), oligodendrocytes are separated into a large main group and a small sub group. The main group of oligodendrocytes is embedded distantly from other cell types, whereas the sub group is embedded proximal to neurons and astrocytes (**Figure 5.11B** and **Figure 5.11C**).
The metabolic states in brain non-myeloid cells are defined by the interactions between genes in the metabolic pathways rather than the differential expression of individual genes. The differentially expressed metabolic genes alone are not capable of separating brain non-myeloid cells into three metabolic states (Figure 5.10A). For example, the expression of Bpgm is significantly higher in oligodendrocytes than other cell types (KS test FDR < 0.001), which is in accordance with previous measurements (Amaral et al. 2016), but Bpgm expressions in neurons and astrocytes are not significantly different (KS test p > 0.05). Therefore, the interactions between metabolic genes need to be systematically evaluated to define metabolic states, which is mathematically implemented by UMAP without prior biological information (Fig 5C) (McInnes and Healy 2018). Although the UMAP and t-SNE embeddings are commonly used in illustrating the cell clusters in scRNA-seq data (Cao et al. 2019; Schaum et al. 2018), interpreting the embeddings and clusters is not straightforward.

Context specific metabolic modeling provides a straightforward way to interpret the metabolic states. The hierarchical clustering results of simulated fluxes in different cell types are consistent with the cell type separation in the UMAP embedding (Figure 5.10B and Figure 5.10C), except the Bergmann glial cells. The fluxes of Bergmann glial cells are more similar to neurons, whereas the embeddings of Bergmann glial cells are admixed with astrocytes. The construction of context specific models and simulation of metabolic fluxes augmented gene expression levels with biological meaningful gene interactions. The model construction procedure uses boolean algebra to determine whether a reaction exists in the context or not, if the reaction requires multiple gene products to catalyze the reaction. The simulation procedure uses the stoichiometry of the reactions, and the steady state assumption connects the fluxes of reactions that share
metabolites. The specific transformation from gene expression levels to reaction fluxes is likely to be non-linear, because most of the expression levels are not linearly correlated with their corresponding reaction fluxes (Figure 5.12B, Figure 5.12C and Figure 5.12D), and the extent of linear correlations is comparable to randomly selected genes (Figure 5.12E). Therefore, context specific metabolic modeling is a viable way to systematically interpret the differences between different metabolic states.
Figure 5.10. Metabolic states within mouse brain non-myeloid cell types. (A) Heatmap of the expression levels of genes in the selected metabolic pathways, including Citric Acid Cycle, Fatty acid oxidation, Glycolysis/Gluconeogenesis, NAD Metabolism, Oxidative Phosphorylation, and Pentose Phosphate Pathway. The bottom ribbon of the oligodendrocytes denotes the two sub-groups of oligodendrocytes. (B) Heatmap of the
flux z-scores of the metabolic reactions in the selected metabolic pathways. (C, D) UMAP embeddings computed using only brain non-myeloid cells and genes in the selected metabolic pathways (C), and the main and sub groups of oligodendrocytes are shown in black and light blue points respectively (D).

Figure 5.11. UMAP 2D embeddings of the transcriptomic profiles of different brain non-myeloid cell types in Tabula Muris dataset. (A) UMAP embeddings computed using the
read counts of all genes and all cells. (B) UMAP embeddings computed using the read counts all genes and brain non-myeloid cells. (C) UMAP embeddings computed using the read counts all genes and brain non-myeloid cells, showing only the oligodendrocytes. The oligodendrocytes are separated into main and sub groups by the first component of UMAP.
**Figure 5.12.** Statistics distributions of the fluxes of brain non-myeloid cells. (A)

Distribution of adjusted rand indices between the k-means clustering results of brain non-myeloid cells by 182 genes in the selected metabolic pathways or randomly selected 182 genes that does not include the selected metabolic genes. The selected metabolic pathways are Citric Acid Cycle, Fatty acid oxidation, Glycolysis/Gluconeogenesis, NAD Metabolism, Oxidative Phosphorylation, and Pentose Phosphate Pathway. A total of 1,000 random samples of genes are used to cluster the cells. The clustering results by random genes are significantly different from the metabolic genes (t-test p < 0.001). (B, C and D) The linear regression p-values (B), R² (C) and FDRs (D) between simulated reaction fluxes and their corresponding mean gene read counts in brain non-myeloid cell types. (E) The minimum linear regression FDR of each random set of 182 genes or the selected metabolic gene set mean read counts to the simulated fluxes in brain non-myeloid cell types.

5.4.5 Simulated metabolic fluxes are reproducible using different scRNA-seq datasets

We constructed cell-type specific metabolic models using the scRNA-seq datasets from Allen Brain Mouse Atlas (ABMA) (Lein et al. 2007), and we simulated the reaction fluxes of multiple metabolic pathways using the constructed cell-type specific models (**Figure 5.12A**). In the results of hierarchical clustering on the metabolic fluxes of various cell-type specific metabolic models, the metabolic profiles of endothelial cells in Tabula Muris, ABMA primary visual cortex (VISp), and ABMA anterolateral motor cortex (ALM) are close to each other. With regard to the metabolic fluxes of astrocytes, oligodendrocytes and macrophages, Tabula Muris and ABMA VISp are similar to each other, but ABMA ALM are distinct from the other two datasets.
We found the simulated NAD⁺ biosynthesis fluxes using Tabula Muris scRNA-seq dataset are reproducible using ABMA ALM and VISp scRNA-seq datasets (Figure 5.12B and
The simulated NAD⁺ biosynthesis fluxes are similar among the common cell types in the three different scRNA-seq datasets. Although the simulated NAD⁺ biosynthesis fluxes tend to have the same ascending order of ABMA ALM < ABMA VISP < Tabula Muris in the same cell type, the order of the simulated NAD⁺ biosynthesis fluxes of different cell types within a dataset is relatively consistent across the three scRNA-seq datasets.
Figure 5.13. Simulated cell-type specific metabolic fluxes using scRNA-seq datasets from Tabula Muris and Allen Brain Mouse Atlas. (A) Comparative Z-score heatmap of metabolic fluxes between Tabula Muris (same TM data as in above right plot). Row pathway color key as plots above. Grouping of cell types for analysis based on cell type
definition from each data source. Abbreviations: ABI Anterior Lateral Motor Area (ALM) and ABI Primary Visual Cortex (VISp). L2-L6 are cortical layers in ABI data. Astro, astrocyte; CR, Cajal–Retzius cell; endo, endothelial cell; oligo, oligodendrocyte; OPC, oligodendrocyte precursor cell; peri, pericyte; PVM, perivascular macrophage; SMC, smooth muscle cell; VLMC, vascular leptomeningeal cell; IT, intratelencephalic; PT, pyramidal tract; NP, near-projecting; CT, corticothalamic. (B) Reproducibility of metabolic model NAD⁺ biosynthesis flux from iMM1415 using CORDA. Results are shown for mean NAD⁺ biosynthesis flux by selected cell types from Tabula Muris, Allen Brain Institute (ABI): Anterior Lateral Motor Area (ALM) and ABI Primary Visual Cortex (VISp) scRNAseq. The points and error bars represent the mean and standard deviation (sd) flux values for NAD⁺ biosynthesis simulated by 20 bootstrapping replicates at 80% of the cells per set. Cell types include astrocytes (Astro), endothelial cells (Endo), oligodendrocytes (Oligo) and neurons indicated by cortical layers 2-5 (L2-L5; L4 is not included in bootstrapping with only 3 cells). Intratelencephalic (IT), pyramidal tract (PT), and corticothalamic (CT). Lamp5: subclass of GABAergic neurons.
Table 5.2. Simulated NAD⁺ biosynthesis fluxes in different datasets. The mean and standard deviation (sd) flux values for NAD⁺ biosynthesis over number of cells (n) simulated by 20 bootstrapping replicates at 80% of the cells per set. Cell types include astrocytes (Astro), endothelial cells (Endo), oligodendrocytes (Oligo) and neurons indicated by cortical layers 2-5 (L2-L5; L4 is not included in bootstrapping with only 3 cells). Intratelencephalic (IT), pyramidal tract (PT), and corticothalamic (CT). Lamp5: subclass of GABAergic neurons.

| Cell Type | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VISP | ABI ALM | ABI ALM | ABI VISP | ABI VSPI...

5.5 Discussion

The biological interpretations of scRNA-seq results are obtained from extensive data analyses, which could take more time than doing the experiments. As the size of scRNA-seq datasets increases rapidly (Svensson, Vento-Tormo, and Teichmann 2018; Rozenblatt-Rosen et al. 2017), the scRNA-seq profiled biological processes become increasingly complex, which involve multiple cell types, tissue types, and organs that are undergoing rapid morphological and molecular changes (Cao et al. 2019; Schiebinger et
214

In such complex biological processes, interpreting the single-cell transcriptomic changes requires integration of relevant regulatory mechanisms of gene expression and cell state transition (Tanay and Regev 2017).

The results of scRNA-seq provide rough measurements of expressed genes in each cell, but the measurements need to be interpreted with extra biological understanding. As biological processes generally involve multiple interacting components, a comprehensive interpretation of single-cell transcriptomic profiles requires a systematic framework that integrates all relevant biological components and their interactions. The systematic frameworks can be developed by mathematically modeling the biological components and their interactions that are comprehensively curated and structured in databases. The previously developed frameworks for interpreting RNA microarray and bulk RNA-seq datasets cannot be directly adapted to interpret scRNA-seq results because of the unique statistical properties of scRNA-seq data and biological purposes of scRNA-seq experiments. Generally, scRNA-seq data have lower signal-to-noise ratio, and scRNA-seq experiments aim to uncover the cellular heterogeneity and identify rare cell populations within cell and tissue types.

We developed a computational method to systematically interpret scRNA-seq dataset in the aspect of cellular metabolism. The method integrates the databases of metabolic reactions (King et al. 2016; Brunk et al. 2018; Kanehisa et al. 2016) to infer metabolic flux variations between different cell and tissue types using scRNA-seq data, through constraint based metabolic modeling (Ebrahim et al. 2013), which is more reliable than associating flux variations with individual gene expression levels (Hackett et al. 2016). In a constraint based metabolic model, the metabolites and reactions are represented as either discrete mathematical graphs (Figure 5.1A) or a system of linear equations
(Figure 5.1B). In the graph representation, each metabolite is a node, and each reaction is an edge, which can be intuitively used for visualization. In the linear system representation, each reaction is an equation with reactants on the left hand side and products on the right, which can be used for simulating metabolic fluxes using optimization and sampling approaches, such as FBA (Figure 5.2) and uniform sampling (Megchelenbrink, Huynen, and Marchiori 2014). The reference model containing all metabolic reactions of an organism can be reduced to represent cell- and tissue-type specific metabolism, according to the scRNA-seq profiled transcription levels of enzymes that are catalyzing the reactions. In order to handle the relatively low signal-to-noise ratio in scRNA-seq data, we agglomerated cells with the same cell or tissue type together to obtain a more accurate transcriptomic profile, and the cellular heterogeneity within the cell and tissue types are evaluated by bootstrapping. The transcriptomic levels are also discretized into five levels to increase the robustness of the modeling and simulation methods.

Computational modeling of cell- and tissue-type specific metabolism using scRNA-seq data is able to guide the design of studies to uncover the regulatory mechanisms of cellular metabolism. Experimental measurements of cellular metabolism are mainly performed by specialized biochemical assays, mass spectrometry, and nuclear magnetic resonance spectrometry (Dunn et al. 2011). As such experiments take a considerable amount of time and resources (Jang, Chen, and Rabinowitz 2018), mathematical modeling of the metabolic fluxes would be able to guide a careful experiment design to facilitate the progression of a study.

The metabolic modeling method that we developed to interpret scRNA-seq results can be improved in multiple ways. The accuracy of the expression level discretization
procedure can be improved by creating customized thresholds for each individual gene rather than using the rank of all genes. The dynamic ranges of the simulated metabolic fluxes can be increased by meticulously setting the bounds of each metabolic reaction according to the enzyme transcription levels rather than only setting the presence and absence of each reaction. The resolution of cell type agglomeration procedures can be improved by dividing single-cells into fine-grained neighborhoods using dimensionality reduction methods.

Also, the developed metabolic modeling method can further be applied to study epigenetic reprogramming and metabolite deficiency. The biosynthesis fluxes of the substrates that are used for epigenetic modifications can be simulated to infer whether the cells are undergoing active epigenetic reprogramming in various differentiation or development processes (Reid, Dai, and Locasale 2017). The biosynthesis fluxes of metabolites involved in cell survival can be simulated to determine whether the cells have sufficient supply of the metabolites (Williams et al. 2017).

The constraint based mathematical modeling strategy of metabolism can be further applied to model other biological processes, through representing relevant biological components and events as networks and linear equation systems. Depending on the complexity of the modeled biological process, different levels of abstraction needs to be applied to ensure the feasibility of modeling procedure. In this way, multiple modeling frameworks can further be integrated to interpret multi-omics datasets.
6 Conclusion

High-throughput sequencing (HTS) technology has substantially expanded biological understanding in various aspects. Whole genome sequencing of multiple species and organisms has extensively characterized the genetic variations between different species and organisms. A myriad of regulatory mechanisms of gene expression have been by various HTS applications that are developed to comprehensively characterize epigenetic modifications of chromatin and DNA, chromatin conformation, transcription factor binding, transcription initiation, splicing, and ribosome mRNA binding. HTS have been applied to profile various cell types, tissues, organs, tumors, and gut microbiota, which has provided insights into the physiology and pathology of metazoa. Coupling HTS with CRISPR-Cas9 genome editing systems enables large scale profilings of the effects of genetic dysfunctions on measurable cellular phenotypes (Dixit et al. 2016; Hoshino et al. 2019), which is promising to generate mechanistic understandings of biology.

Exploiting HTS, we obtained new insights in transcription initiation. We identified promoter region sequence patterns that can determine the biochemical properties of RNA polymerase in transcription initiation, including transcription start site site selection, transcript yields, transcriptional slippage, and utilization of non-canonical nucleotide for initiation. Also, we find that RNAP scrunches DNA prior to RNA synthesis for TSS selection in transcription initiation.
We also developed various computational methods to explore scRNA-seq datasets. The methods are efficiently implemented in Python programming language to utilize parallel processing, which are organized into a Python package. The package is distributed freely on Python Package Index, which can be easily installed with Python programming language version 3.5 and 3.6 runtimes. The package is reliable and easy-to-use, and we also provided detailed documentation about the usages. The modular design and comprehensive tests of scedar facilitate customization for specific use cases, extensibility for incorporating analytical procedures, and for future refactoring to further improve scalability.

In addition to general data exploration, we developed a method to systematically model cellular metabolism using scRNA-seq datasets. The simulated metabolic reaction fluxes are reproducible using different scRNA-seq datasets, and they showed significant linear correlation with experimental measurements. We further modeled the metabolism of mouse brain non-myeloid cell types, and we found three cellular metabolic states among the cell types. The flexibility of the scRNA-seq metabolic modeling method facilitates its future adaptations to identify cellular states that are undergoing rapid epigenetic reprogramming or deficiency of essential metabolites.


Hicks, Stephanie C., and Rafael A. Irizarry. 2015. “Quartro: A Data-Driven Approach to Guide the Choice of an Appropriate Normalization Method.” Genome Biology 16 (June): 117.


Kim, Sang-Young, Bruce M. Cohen, Xi Chen, Scott E. Lukas, Ann K. Shinn, A. Cagri


Qiu, Xiaojie, Andrew Hill, Jonathan Packer, Dejun Lin, Yi-An Ma, and Cole Trapnell.


and Reviews in Molecular, Cellular and Developmental Biology 39 (2).
https://doi.org/10.1002/bies.201600193.