Eberhard Karls Universität Tübingen
Mathematisch-Naturwissenschaftliche Fakultät
Wilhelm-Schickard-Institut für Informatik

Bachelor Thesis Bioinformatics

Time Series Data Visualization in Escher

Christoph Blessing

September 8, 2017

Reviewer

Jun.-Prof. Dr. Julian Heinrich
Applied Bioinformatics Group
Sand 14, 72076 Tübingen
University of Tübingen

Mentor

Dr. Andreas Dräger
Applied Bioinformatics Group
Sand 14, 72076 Tübingen
University of Tübingen

Blessing, Christoph:
Time Series Data Visualization in Escher
Bachelor Thesis Bioinformatics
Eberhard Karls Universität Tübingen
Period: May 9th, 2017 - September 8th, 2017

Abstract

Modern biological research has arrived in the age of big data. Consequently, new discoveries heavily depend on specialized software for analysis. To understand biological processes it is not sufficient to consider a model in stasis but also to analyze the systems dynamics. In the field of pathway-based visualization the web-based tool Escher offers a powerful program for visual and statistical management of large data sets in the context of metabolomics, transcriptomics, and flux visualization. So far, it could not accept time course data.

In this thesis Escher was extended to enable a time series data visualization including animation. By implementing new features in the existing code base, data can now be contextualized over a time course or in between different conditions. A user interface was designed to control the data playback animation (with or without interpolation) in a time series or in pairwise comparison of data sets. This extends Escher with essential tools for analysis of time series data in the context of metabolism and can lead to new findings in biological research.

The source code is available at `https://github.com/christophblessing/escher` (MIT license).

Acknowledgments

I would like to thank Prof. Dr. Julian Heinrich, Dr. Andreas Dräger and Dr. Zachary A. King (UC San Diego) for providing me with this interesting topic and guidance during my thesis.

Furthermore, thanks to all my friends and family including Vanessa Murek, Lea Buchweitz, Jennifer Bödker and Thomas Riedlinger for proofreading my work and company during desperately needed coffee breaks.

Contents

List of Figures

List of Abbreviations

ADP	Adenosine diphosphate
ATP	Adenosine triphosphate
API	Application Programming Interface
CSV	Comma-Separated Values
D3	Data-Driven Documents
DOM	Document Object Model
GUI	Graphical User Interface
IDE	Integrated Development Environment
JSON	JavaScript Object Notation
SBML	Systems Biology Markup Language
SVG	Scalable Vector Graphics
XML	Extensible Markup Language

Chapter 1

Introduction

Time is the constant progress of events we arrange in the order of past, present and future. The world we live in is constantly evolving and changing in this manner. To study biology, the science of living organisms, it is of paramount importance to understand not only static systems but their dynamics, because only in change there is life.

In the past, the approach to gain deeper insight in living organisms was to study individual components with the help of molecular biology, biochemistry and genetics. But even a fundamental understanding of these disciplines did not fulfill the pursuit to reveal genotype-phenotype relationship [BMKP14]. The genotype describes all the genes of an organism, the phenotype the observable physiological and biochemical properties. It is a long desired goal of scientists to understand this relation. To address system wide interactions and study behavior of an organism, systems biology arose [Kit02].

In this context, visualization can help to grasp the overwhelming complexity of how organisms interact with the world around them. It is of interest because different conditions, changes and prediction of behavior can be simulated with different techniques [BBDW14].

In present times next generation sequencing and "omics" technologies have lowered the cost of data generation and the biological sciences have now arrived in the age of big data [BMKP14]. While manual analysis became too expensive and time-consuming, the focus shifted towards computational methods. The immense number and huge size of the resulting data sets brought the classic way of biological research to its limits. To make future discoveries, the bottleneck is now data analysis. New computer supported approaches are made with advancing processing power and sophisticated algorithms. Today there are a lot of programs that are specifically designed for data analysis via visualization like Omix, Cytoscape and CellDesigner [DNW13, SMO$^+$03, FMKT03]. Escher, a web-based tool for metabolic pathway visualization, is joining the ranks [KDE$^+$15]. So far, it can handle single data sets or pairs of data related

to metabolites, reactions, and genes. It can be used to visualize results in the fields of metabolomics, calculated fluxes and transcriptomics. However, it is not able to visualize time series data, yet. This thesis introduces a new version of Escher with extended features to accept, visualize and animate time course data.

This thesis is structured into six different chapters. After this introduction, Chapter 2 covers all the relevant basics of the biological and technical background. This will also contain an explanation of the software tool Escher. The following Chapter 3 with results will cover everything from a brief introduction to the used techniques to the overall design of the program and a detailed documentation of the extension. In the end of this chapter there will be application examples to show the new implemented features. Materials used will be listed in Chapter 4. In Chapter 5, the discussion, the relevance of this work and further improvements will be explained followed by a conclusion and outlook in Chapter 6.

Chapter 2

Basics

In this chapter the biological background as well as the basics of systems biology and visualization of biological networks are explained. Furthermore, it contains a introduction to the features of the software Escher.

2.1 Biological Background

Most of the data generated in biological studies are either taken from direct measurements (primary data), derived from such by prediction (secondary data) or a combination of both [AKK+09]. This data is usually analyzed by interpretation as a graph structure.

Next-generation Sequencing and "Omics" Technologies

Genetic information is encoded in DNA, which is a sequence of the four different bases adenine, guanine, cytosine, and thymine. The order in which they are aligned carries the blueprint of living organisms by defining its structure and function. It is a major goal of life sciences to understand this genotype–phenotype relationship [BMKP14].

Recently groundbreaking accomplishments in sequencing technologies revolutionized the biomedical research by providing huge amount of inexpensive data which created a new need for data analysis software [XZAY11]. There are a lot of different next-generation sequencing methods. All have in common sequencing millions of small DNA fragments in parallel. To obtain the whole DNA sequence, the fragments are put together using bioinformatic methods [BT13].

The term "omics" denotes to technologies and methods referring to genes, mRNA, proteins and metabolites (genomics, transcriptomics, proteomics and

metabolomics) [HK11]. The integration of these technologies is called "systems biology" which is described in the next section.

2.2 Systems Biology

Due to the vast complexity of biological beings is it not sufficient to consider only isolated parts of cells and their characteristics but to study the structures and dynamics on a system level. Genetics, biochemistry and molecular biology analysis lays the basis through the collection of comprehensive data of the individual components. Systems biology is the approach of bringing together this detailed information to an understanding of the organism as a whole system [BMKP14].

Recent advances in molecular biology such as next-generation sequencing and "omics" technologies brought more possibilities to examine mechanisms in organisms [KDE+15]. These mechanisms come from interactions between genes, proteins, reactions and metabolites that form the systems behavior. Data, especially gained from genome sequencing and high-throughput measurements, allow a comprehensive comparison in detail [Kit02]. Especially the observation of a system over a time course is of great interest.

It is important to understand three crucial concepts of a biological system and how they are interpreted in a model. Interaction of components leads to the network structure which is related to their function. Its dynamics are based on the mechanisms that control the state of the system. The resulting model is a representation of a biological phenomenon.

Supported by experimental data, a computable set of hypotheses and assumptions is the first step of creating the model. Dry experiments such as simulation need to be verified by wet experiments. In Figure 2.1 the process of hypothesis-driven research in systems biology is shown. The focus in this thesis lays in the computational part, especially in modeling, simulation and system analysis, consequently can be categorized in "dry" experiments.

Systems Biology Markup Language

Working in the area of computational biology often requires to combine work done with different software to process the data. When simulation software becomes outdated, models created with them are often no longer usable [DP14]. To address these problems, a widely accepted data standard to exchange biological models is the Systems Biology Markup Language (SBML). It is describing biological models with information about signaling pathways in cells, gene regulation and metabolic pathways by using Extensible Markup Language (XML) for storage [HFS+03]. In SBML Level 3 [HFS+03] it is possible

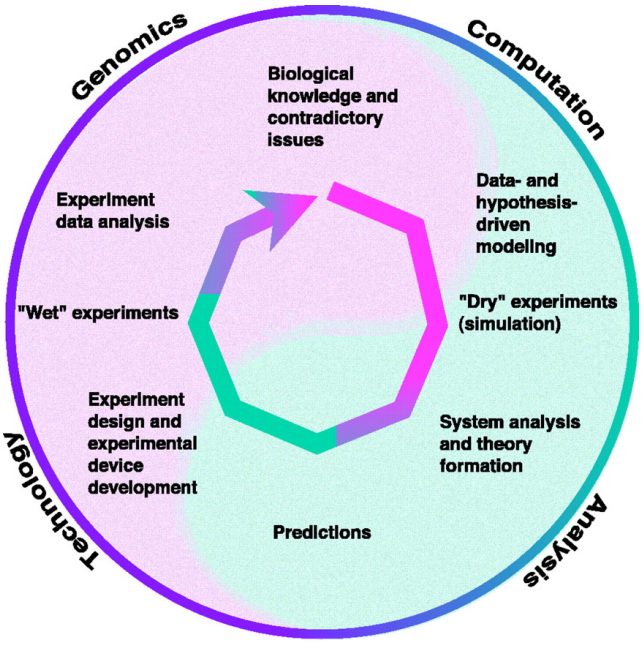


Figure 2.1: The idealized circle of hypothesis-driven research in systems biology. The starting point of the research cycle is based on the issues arising in biology. Based on the biological knowledge a model is created to represent the parts of the hypothesis that can be further examined computationally by 'dry' experiments like simulation. This thesis is a contribution to this part of the circle. Followed by further analysis, the model is either rejected if it contradicts established evidence or used to make predictions that are tested in wet experiments. If a model passes all the stages and is not contradicted through experiments it is seen as consistent with experimental facts. Source: [Kit02]

to store layout information [GRS+15] which is of great importance to extract knowledge of a visual representation [GRSW06].

2.3 Visualization

Biological network data is usually visualized in a graph structure and can be used in various contexts. The most central types in systems biology research are gene-regulatory networks, protein interaction networks and metabolic networks [AKK+09]. In this thesis only the last one is dealt with and its principles will be examined further.

In general, graphs are one of the most important data structures in computer sciences. They can model a lot of problems in an abstract way. To represent a structure a static graph $G := (V, E)$ is made of objects known as *vertices* V and *edges* E to show relationships between them. A dynamic graph is used to show changes of states over time or in between conditions. It

is defined as a sequence of static graphs $G_s := (G_1, G_2, \dots G_n)$ with G_i, the static graph representing a given point in time $i = 1 \dots n$ [BBDW14].

Biological networks are commonly represented as directed graphs to show pairwise relationships in biological pathways. Metabolic networks are a representation of how metabolites are transformed into other metabolites. Typical for metabolic networks is the representation as a 'node-link' diagram. It is easy to follow the pathway and identify relationships between components [GW12]. *Edges* characterize the reaction flow. *Vertices* are typically metabolites but can also be enzymes, which catalyze the chemical conversion from one metabolite into another. Thus, the node set V is divided into the set V_1 of metabolites and the set of reaction vertices V_2. Resulting in a bipartite hypergraph $G = (V_1 \cup V_2, E)$. An illustration of such a graph can be found in Figure 2.2 showing the first step of the glycolysis after phosphorylation with all reaction and metabolite components to show the process.

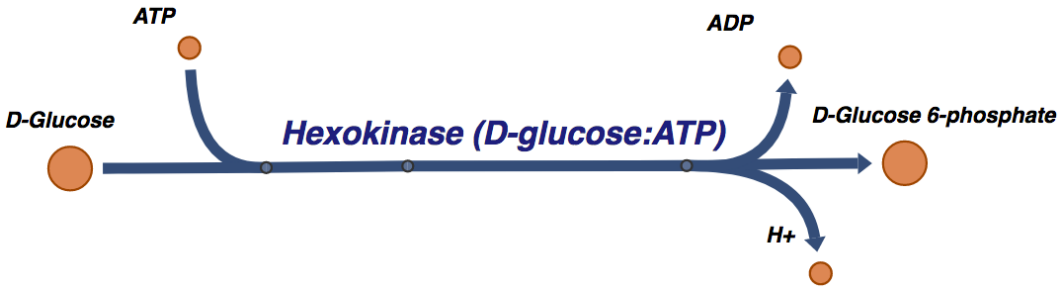


Figure 2.2: Example of a node-link diagram to visualize the metabolic pathway relationships of the first step of glycolysis after phosphorylation. D-Glucose (metabolite node) on the left side of the reaction is catalyzed by the enzyme hexokinase (reaction node) relying on ATP (metabolite node). Glucose 6-phosphate is formed while hydrogen (H) and ADP are split off (all metabolite nodes). The reaction edges indicate the irreversibly of the reaction. All together form a bipartite hypergraph. For this picture Escher was used with the *Saccharomyces cerevisiae* iMM904 model and map [MPH09].

Pathway-based visualization is a tool for the analysis of biological processes and their change over time. Pathways show information flow encoded by the directions of the edges [HK16]. Normally, the speed of a reaction is indicated by the thickness of the reaction edge represented as an arrow. Metabolites are commonly drawn as circles and different states are visualized as size or color of the nodes. Reconstruction of the biochemical reaction and metabolic networks is done by systematically analyzing the networks flow, which is the state at the network [BMKP14].

Usually, additional data is mapped onto either nodes or edges to get a visual representation of the primary or secondary data. For a deeper understanding of complex biological systems structure and dynamics of the network

are important, e.g., to spot interesting correlation patterns. Such patterns can be a set of nodes that are close together in the graph but do not show any connection with the data represented on the nodes [AKK+09]. This point of interest can be a starting point for further research.

Another way to come upon results is to explore the same network in different states. These can either be the same system at various time points, under other environmental conditions (pressure, temperature, pH value, concentration of substrate) or measurements on healthy vs. diseased organisms. Mathematically this results in a set of graphs $G_1, \dots G_n$ and points of interest can be discovered by highlighting differences between these networks or in a time series comparison. For example research done this way may help to identify a disease-specific pattern [AKK+09].

Flux balance analysis

Fluxes describe the distribution of mass and signal flows in a network. They also can be examined through considering uncertainties in the data [VHK+12]. Through a time series of measurements the change over time of these fluxes can be studied. A biological network can contain such large numbers of flows that information must be filtered to obtain a network of observable size. This reduction to the main paths can be made by only considering the paths that have a respectable flow or have a statistically significant overall share of the entire flow. To represent quantitative and qualitative properties the edges of the graphs can be weighted. Visualization can be done by encoding the information in edge widths and colors. For instance, points of interest can be if a certain substructure in the graph has a large part of the flow at a given point of time that shift into another part at a later point.

To examine the information in these models, the most commonly used approach is the flux balance analysis (FBA). Such a model contains all the metabolic reactions and genes encoding the enzymes catalyzing them. FBA calculates the flow of metabolites in the network represented by the model. Examples of applications are to predict the growth rate of an organism or the production rate of a metabolite important in biotechnological contexts [OTP10]. The verification of predictions based on such a model has to be made in the same way they are created which is by analysis of experimental data.

2.4 Escher

Data sets in biological sciences have become too complex and large to be analyzed by hand so a demand for special programs arose. To bring together statistical and modeling methods the demand for data visualization software

has gained more and more interest recently. In the context of metabolism such a tool is the web based application Escher [KDE+15]. This section is an introduction to the features of Escher and why it is so powerful and flexible.

Core features of such programs are a visual representation of processes in the pathway in a biochemically correct way. Navigation through the whole visualization as well as design and customization of the pathway maps should be possible. Appropriate representation of different data types, like color or size, is essential for a visualization one can make sense of. Storage, import and export features for sharing with other programs as well as an API for utilization in data analysis pipelines is also of great importance. Unlike many other web applications, the web-based open source tool for metabolic pathway visualizations Escher satisfies all these key features. Escher is a tool for visual and statistical management of large data sets in the context of metabolism. There is no need to download Escher to use it and it does not need any plug-ins, so it is immediately ready for use and also platform independent. Through performance increase of modern computers in recent times its performance is comparable to desktop programs for a lot of applications [KDE+15].

Visualizations made with Escher can be adjusted and extended by the user. It also contains export functions for sharing as well as a feature to be embedded in web pages. The user can design pathway maps quickly, because Escher offers a drawing tool which suggests the next steps in the pathway based on the genome-scale model. Escher is able to visualize data of reactions, genes and metabolites in a biological pathway. An example of its customization and visualization capabilities are shown in Figure 2.3. All this makes it worthy a target for further development.

So far, Escher can display a single data set or a comparison of up to two data sets. High level goals to achieve were set to extend Escher to accept time course data for metabolomics, transcriptomics, and flux visualization. New standards for data storage needed to be realized. The user interface was changed to give the user control, e.g., to switch between data sets representing different conditions of the network. Finally, an animation of the time course was created.

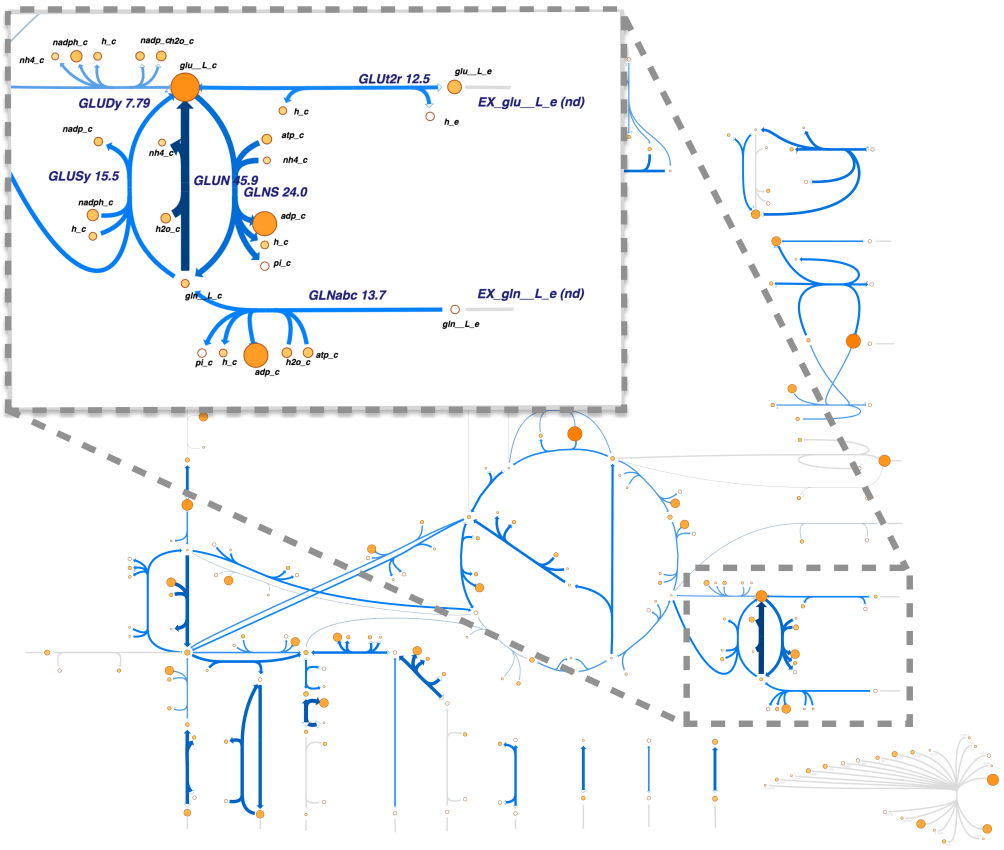


Figure 2.3: Screenshot of the *E.coli* core map in Escher. It is used as a tool to visualize the *E.coli* core map, gene data iJO1366 and metabolite data [OFP10, OCN+11]. In the background the map is shown in little detail to get an overview of the whole metabolic pathway. In the upper left corner box all the details are displayed to emphasize its customization features. The coloring and size of metabolites show the different concentrations of metabolites in orange. Reaction data differences are visualized as a light to dark blue. This is a static representation of the loaded data sets. In this thesis modifications were made to give a time series data visualization by animating differences of data sets over a time course. The metabolite and reaction color and size will change in an animation.

Chapter 3

Results

Escher is a web-based tool for visual and statistical managing of large data sets in the context of metabolism. However, it is only able to display static information of these sets so far. The objective of this thesis was to implement a time series data visualization. This chapter contains a detailed description of all the steps necessary to reach this goal.

Extending an existing program brings different challenges than starting from scratch. One aim was to use as much existing code of Escher as possible. This way it is less likely to have bugs and leads to a more stable code. The majority of classes remained unchanged. Modifications were done in the classes `Map`, `data_styles`, `draw` and `Builder`, where most of the new code is written. A new class was written containing the Graphical User Interface (GUI) for time series visualization called `TimeSeriesBar`. It contains all the logic for data playback and user interface elements.

In the end the stable release will be part of the core Escher program and will follow its Semantic Versioning guidelines [KDE+15]. That means functionality is added without breaking the Application Programming Interface (API) and backwards compatibility will be ensured. The goal of achieving a minimum viable product was aimed at. In the released online version further documentation and testing will be done.

3.1 Software Design

This chapter contains detailed information about the implementation of the code that was written as an extension of Escher.

The biggest challenge in the beginning was to get familiar with the existing code base of Escher. Its nature as a web application placed a lot of value on performance. It follows a programming style that uses no global variables to save memory and tries to save as much computing time as possible using

Boolean checks. First steps were taken in understanding how the program reads and processes data. From user interaction, to data validation, to data processing and statistics the path led to visualizing of reactions or metabolites and calculating scales.

The original version of Escher was able to get information out of a maximum of two data sets that are used to visualize the difference between them. The data input can be in the Comma-Separated Values (CSV) or JavaScript Object Notation (JSON) format. One key element to enable time series data visualization in Escher was making the program able to process data that contain more than two data sets and calculate the scale across all.

Changes were made in the classes `Builder` and `data_styles` and Escher was extended to hold as many data sets as requested. A GUI was designed to switch from single to difference data in form of the `FindBar` style element centered in the screen.

Specification of data set names by the user allow determining time points, conditions or states, e.g., aerobic or anaerobic, of the data. Also, a time series can be on a nonlinear scale. The user can specify this by setting the names in a certain way and so it is possible for Escher to extract the numerical time values from the data labels. This will be important for playing back time series as a continuous animation. This was quickly implemented in CSV, because the needed information can be extracted from the column headers, which so far were ignored. However, JSON data needed a new format to include this information. Several possibilities were considered and after further testing it will be decided which one will find use in the released version.

All these steps were necessary to implement a time series visualization in Escher. The next focus was on data playback and interpolation.

3.1.1 Data Playback and Interpolation

In an experiment often only selective measurements of a phenomenon are taken point by point, e.g., in biology a metabolite concentration over a specific time span. There is no mathematical function given for the data that lays in between. Interpolation is a method to approximate these unknown data points and creates a continuous curve that is ideal to show a trend from one data point to another. If the dynamical change in the flow over time is of interest data interpolation gives a smooth animation [AKK+09].

Data interpolation in Escher was implemented with the Data-Driven Documents (D3) `transition` function which uses the existing value of the object and interpolates to the new value specified in the next data set [BOH11]. In this case the value of a metabolite circle or flux of a reaction arrow edge. Using a transition duration gives a smooth animation blending into the next data set without creating a lot of data overhead. The idea is to change the circle radius

and color of metabolite with the change of concentration over the time series and in between time points. Changes in reaction and gene data are visualized as stroke color and thickness. An illustration of different states in a network can be found in Figure 2.3. D3 calculates the data points with the `easeLinear` function using a linear interpolation.

The data playback itself can be of a linear time course, or, if the time points specified in the data set as in section for data standards 3.1.3, as a nonlinear time series. Using the built-in JavaScript function `interval` repetition of the animation until further interaction is enabled. An illustration of a time course can be seen in the application example in Figure 3.2. It shows a subset of the metabolic network in two different states, whereby the transition between them is interpolated. Also, example videos of animations can be found under further material A on CD.

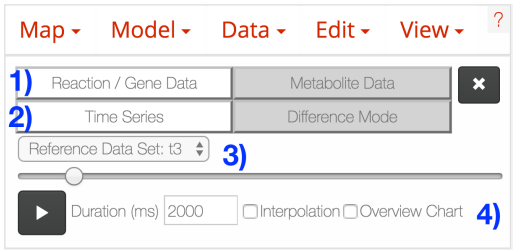
3.1.2 Graphical User Interface

The objective was to implement a simple and functional user interface. In later versions of Escher the Bootstrap library will not be supported. This is why for this thesis a basic GUI was designed so in the long-term fewer changes have to be made when the library is removed. Using only standard HTML5 elements made this achievable. The new interface is located directly under the standard menu bar of Escher.

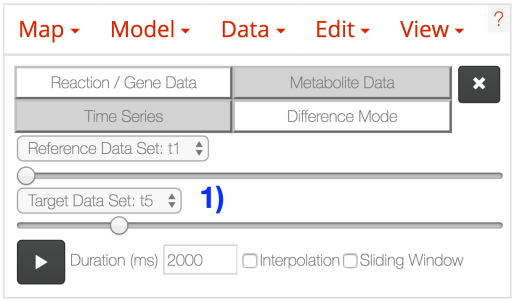
Key elements in the tabbed layout are the two top buttons to choose reaction, respectively gene data, or metabolite data to be handled (Figure 3.1a, 1). These buttons are only active when a matching data set is loaded correctly. The next row of buttons are for choosing between time series animation and Difference Mode (Figure 3.1a, 2). Based on the choice different control elements are shown and the selected button has a white background. This gives a tabbed layout with clear indication which type of data is handled how with the controls. Pressing the play button it offers a linear, or if specified in the data set, a nonlinear time course animation with or without interpolation.

Both modes contain sliders and drop-down menus (Figure 3.1a, 3). A drop-down menu has two functions: It works as a label to display the name of the currently selected data set and to choose between the data sets loaded in. A Slider also lets the user switch between the data sets and acts as a progress bar to give an indication which data set is currently displayed in the time series.

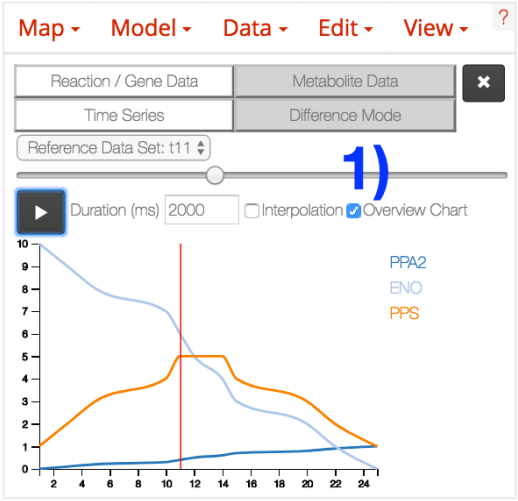
The check box labeled “interpolation” activates specific D3 functions that compute a smoother transition between the data points for a continuous animation. To set the total duration of the animation in milliseconds, an input field takes the desired value (Figure 3.1a, 4).



(a) GUI in Time Series Mode



(b) GUI in Difference Mode



(c) GUI in Time Series Mode with Overview Chart

Figure 3.1: The user interface in three different modes. All have a common input field, labeled "Duration", where the duration of the whole animation can be set and play button to start the animation. If the "Interpolation" check box is activated, a smoother animation with interpolation is created.
Time Series Mode: The user interface in Time Series Mode shows a target slider and drop-down menu, which also indicates the selected data set as a label, to select a data set to visualize. Pressing the play button the slider moves as a indicator which data set is currently displayed.
As seen in 3.1c if the check box "Overview Chart" is selected Escher draws a chart of the loaded data set in Time Series Mode. When playing in a time series a red status bar indicates the current animation progress. Lines for every different Reaction (respectively gene, metabolite data) visualize the time course of the data with a label in the same color to identify the reaction or metabolite value.
Difference Mode: In Difference Mode a second slider and drop-down menu are added to the interface for setting a target data set. This target data set is used for comparison to the reference according to the function set in the settings (difference, log or log-fold). Pressing the play button the target data set remains static while the target moves one step at a time. If the check box "Sliding Window" is activated, the reference moves at the same time the target moves to create a time series animation in a slinging window.

Time Series Mode

Time Series Mode, like shown in Figure 3.1a, offers one slider and a drop-down menu, both labeled as reference (Figure 3.1a, 3).

Check box "Overview Chart" draws a basic line chart that plots all data sets in the time course (Figure 3.1c, 1). Lines are in the same color as the label to indicate the relation. This way of visualization offers an overview of all processes and a tool to spot particular points, e.g., a maximum or minimum respectively a drastic change in the course. Together with the animation a red indicator line follows the course. Figure 3.1c gives an example how the chart is displayed with a data mock-up.

Difference Mode

In a set of data it can be of particularly importance to display the difference of a reference value to all others in a data set. For instance, in a time course the first measurement vs. any other given time point has to be compared for generating results. Another application could be examining the differences between data taken from a healthy patient compared to samples of a sick.

For these kinds of tasks the Difference Mode contains two sub modes and a second slider as well as drop-down menu are added to the interface for setting a target data set (Figure 3.1b, 1). First the reference is locked on the position chosen by selecting a data set as reference with either slider or drop-down menu. Pressing the Play button Escher goes through all data sets and linear displays the difference between the current two sets. An illustration can be found in Figure 3.2a.

Selecting the sliding window check box, which replaces the chart overview check box of the single mode, the user sets up a reference and target point. This distance of data sets is kept for the whole time course and the difference is shown accordingly. An illustration can be found in Figure 3.2b.

```
reference: [0] 1  2  3
target:     0 [1] 2  3
----
reference: [0] 1  2  3
target:     0  1 [2] 3
----
reference: [0] 1  2  3
target:     0  1  2 [3]
```

(a) Difference Mode. While the reference data set remains static, the target moves one step at a time. The difference value between the data points according to the setting is visualized.

```
reference: [0] 1  2  3
target:     0 [1] 2  3
----
reference:  0 [1] 2  3
target:     0  1 [2] 3
----
reference:  0  1 [2] 3
target:     0  1  2 [3]
```

(b) Sliding Window. Reference and target data set move one step at a time in a priorly set distance. The difference value between the data points according to the setting is visualized.

Figure 3.2: Illustration of Difference Mode in the two sub modes with and without sliding window.

3.1.3 Standards for Data Storage

Three different types of data were designed as part of this thesis. After further testing it will be decided which type of data is the most practical for the user. For each data type an example data set is provided in the further materials section A on CD.

Type 1 One array containing name of data set and dictionaries with identifier and data.

Type 2a Two arrays. First one containing name of data sets, other array of dictionaries with identifier and data.

Type 2b Two dictionaries. First one containing name of data sets as arrays, the other is array of dictionaries containing identifier and data.

If the user decides to write data in CSV format, the header row contains the data set names (or time points) and the first column as the identifiers of genes,

metabolites or reactions. After that data is written accordingly in rows and columns.

The new version of Escher is designed to automatically recognize nonlinear time scales. For this feature the user must name the data sets in a specific way. The data set name has to begin with the letter `t` and followed only by a number. The number represents the time when the measurement was taken. For instance, `t0` will be interpreted as t_0, `t10` as t_{10}.

3.1.4 Data Preparation

This section describes what kind of up-front data cleaning and manipulation the user should do to get the best visualization results in Escher. In the ID column of CSV data sets and JSON name strings it is important to put in the descriptive names or BiGG IDs [SPCP10]. Otherwise, Escher will not be able to correctly map the values onto the network. For a better understanding a set of mock-up data files can be found in the further materials section A on CD. Normalizing data is often useful for a time series visualization as well as to transform each value by the logarithmic function. After this preparation high and low values, as they often occur in biological data, will be visualized more uniformly. However, data without prior normalization works in Difference Mode with the built-in $\log_2$ or fold change functions found in the Escher settings.

3.2 Application Examples

Escher provides all the necessary settings for a time course animation. For this application example a comprehensive metabolite data set collected from the human platelet (PLT) was used. During storage over time a condition called PLTs storage lesion (PSL) is developed and was examined further in course of a comprehensive metabolomic study [PSR+14]. The model is the biochemical reaction network of the PLAT metabolism iAT-PLT-636 and constructed to examine metabolic signatures for aspirin resistance [TRB+14]. For an adequate time series visualization the data and default settings were adjusted as described in the following section. This gives an insight how the extension of Escher made in this thesis is applied in practice.

Data Preparation

The data was prepared with the following R [Tea14] script which also can be found under further materials A on CD:

```
# Input and Preparation
data <- read.csv(file="/Users/Christoph/Documents/Studium␣
    Bioinformatik/07␣SoSe␣17/Bachelorarbeit/PLT/PLT_ALL_time_in_
    days_escher.csv",head=TRUE,sep=",")
matrix <- as.matrix(data)
dataMatrix <- matrix(matrix, ncol = ncol(data), dimnames = NULL
    )

# cut off the header row
dataMatrixWithOutNames <- dataMatrix[,-1]

# keep data set names
firstRow <- names(data)
dataSetNames <- strsplit(firstRow, "␣")
dataSetNames <- unlist(dataSetNames)

# cut off identifier
identifier <- dataMatrix[,1]
identifierList <- list(dataMatrix[,1])

# Data as vector for function
dataVector <- c(dataMatrixWithOutNames)
x <- as.numeric(dataVector)
# Logarithmic function
x <- log10(x)
# Data normalization
normalized = (x-min(x))/(max(x)-min(x))

# put together identifier and data
mData <- c(identifier,normalized)

m <- matrix(normalized, nrow = nrow(data), ncol = ncol(data)-1,
     dimnames = NULL)
df <- data.frame(identifier, m)
colnames(df) <- dataSetNames

# Output
options(scipen = 999)
write.table(df, file="normalized_data.csv", row.names=FALSE ,
    sep=",")
```

How to use Escher

The modified version can so far only be run locally. The source code is provided in the further materials section on CD or can be downloaded from `https://github.com/christophblessing/escher` (MIT license).

For further usage Escher offers a tutorial in its documentation which can be found at `https://escher.readthedocs.io/en/stable/`. The CD also provides a test version. To start an Escher development server the following script must be run:

```
./node_modules/webpack-dev-server/bin/
webpack-dev-server.js --hot --inline --color --progress
```

After that, Escher can be tested under `http://localhost:8080/`. The visualization parameters can be adjusted in View » Settings.

Time Series Animation

Under further materials A on CD an example video named `escher_application_example_PLT_time_series.mov` can be found showing a time series animation of the PLT map and data set. For the video a reduced map was used due performance issues with screen recording (hardware specifications under chapter 4). The videos first shows how map and data set are loaded. Followed by adjusting the settings and showing a time series animation with interpolation. With the following settings it can be reproduced:

1. Map » Load Map JSON `PLT_Redrawn_reduced_size.json` from example_data from further materials on CD
2. Data » Load metabolite data `normalized_data.csv` from example_data from further materials on CD
3. Go to View » Settings » Metabolites and enter the following:
   - Min to a white color and size to 10
   - Max to a bright red color and size 80
4. Display first data set by either choosing t_0 form drop-down menu or set the slider to the far left
5. Set Duration to 5 000 (ms)
6. Select the check box “Interpolation”
7. Press play

For a quick demonstration of a time series animation the video `escher_time_series_animation_TCA.mov` shows Escher visualizing the TCA cycle with metabolite data.

Time Series Animation Overview Chart

A demonstration video of the overview chart can be found as `escher_overview_chart.mov`. It shows a data mock-up of metabolite data on several reactions showing how the chart gives an overview of the time course.

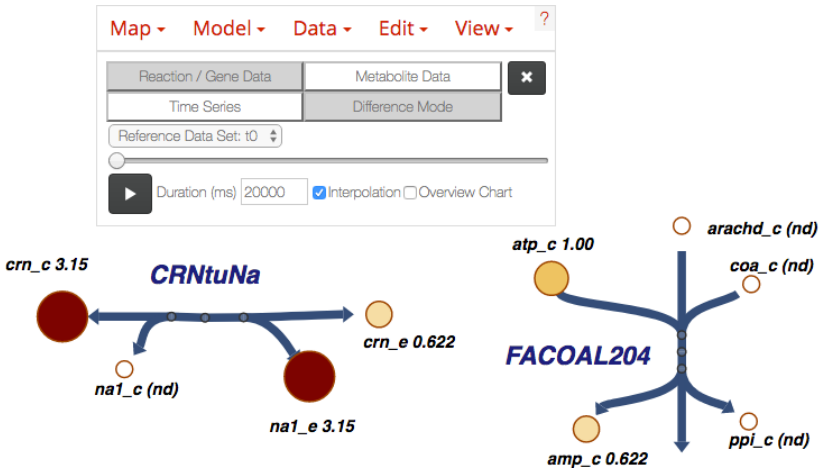
Difference Mode Video

An example of reaction data can be found as `escher_time_series_and_difference_mode.mov`. It shows a data mock-up of the endolase reaction. First in Time Series Mode, followed by an demonstration of both Difference Mode with and without sliding window.

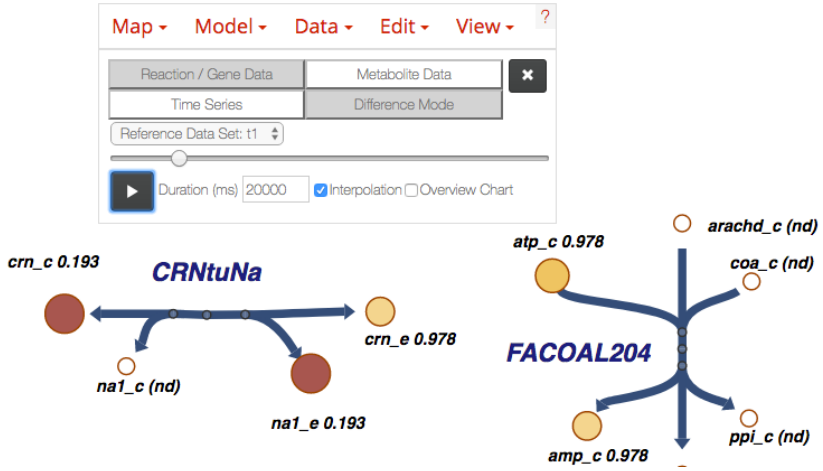
Settings for Example Figure

The following description is how Figure 3.3 was produced. The changes in metabolite concentration are shown as a transition from beige for a low concentration to orange for the median value. A high concentration is visualized as a deep red color and also the change in size is concise. The following settings were used:

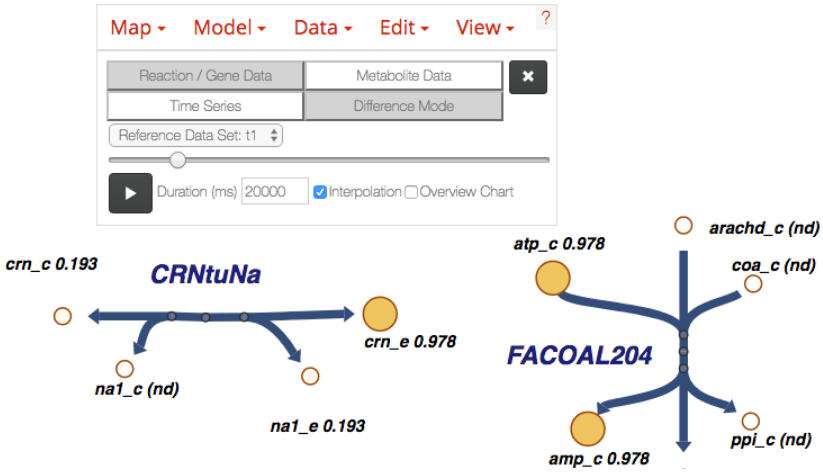
1. Map » Load Map JSON `PLT_Redrawn.json` from example_data from further materials on CD
2. Data » Load metabolite data `PLT_ALL_time_in_days_escher.csv` from example_data from further materials on CD
3. Go to View » Settings » Metabolites and enter the following:
   - Min to a beige color and size to 20
   - Select median as divider and set to a orange color and size 30
   - Max to a dark red color and size 40
   - Metabolites with no color in white and size 10
4. Display first data set by either choosing t_0 form drop-down menu or set the slider to the far left
5. Set Duration to 20 000 (ms)
6. Select the check box “Interpolation”
7. Press play



(a) Time point 0.



(b) Time point 0.5



(c) Time point 1.

Figure 3.3: Time series data visualization application example of PLT data [TRB+14, PSR+14]. The upper picture 3.3a shows the data of the first time point, 3.3c of the second. 3.3b is not included in the data and is the result of the interpolation between 3.3a and 3.3c. Note the metabolite crn_c for the change in the time course. The scales are set to display a high metabolite concentration in a red color, medium in yellow and low concentration as a white color.

Chapter 4

Material

In this section materials are listed that are fundamental for developing in Escher. Only technology that is already used to build the tool was used to extend it. The Integrated Development Environment (IDE) of choice was WebStorm in Version 2017.2 by JetBrains and was used for all programming tasks on a MacBook Pro 13-inch, Early 2011. For testing and debugging the developer tools of the Google Chrome browser was used in Version 60. Video recording was made with QuickTime Player Version 10.4.

4.1 JavaScript and Libraries

Escher is primarily written in JavaScript and extended with libraries Bootstrap (design template library), JQuery (Document Object Model (DOM) manipulation library) and D3. For this thesis ECMAScript Version 5.1 was used.

4.2 D3

Data-Driven Documents (D3) is a JavaScript library for creating data visualization in web browsers [BOH11]. It is already used to visualize all graph elements in Escher and is now used to visualize the time series animations.

4.3 JSON

JavaScript Object Notation (JSON) is a human readable data structure and used to store data for metabolite, reaction and gene data for Escher. Further specifications can be found in Subsection 3.1.3 as well as example data in further materials A. Escher maps are also stored as JSON files. Using the SBML with layout extension maps are drawn on the canvas.

4.4 CSV

Comma-Separated Values (CSV) data format is written in plain text which lets the user store data for metabolite, reaction and gene data similar to JSON. For further specifications see Subsection 3.1.3 and example data can be found under further materials A.

Chapter 5

Discussion

In this work the basis for a time series data visualization in Escher is set with implementing all the key features needed. The extension of the program offers fundamental tools for an analysis of time series and difference data. Furthermore, with the content of this thesis a solid foundation was laid for further improvements on visualization techniques in Escher.

5.1 Escher

By request this new version of Escher can draw an overview chart (Figure 3.1c) of the whole time course offering a useful tool to spot interesting points in the time series. Still, improvements can be done. The user may want to customize the scales and interpolation techniques of the graph to examine the data further. A time course data set can contain so many reactions that it would be useful to have the choice to visualize only specific ones. One key feature of Escher is to export and share metabolic pathway maps. With further improvements it could be possible to export a customized chart as Scalable Vector Graphics (SVG) and even embed it in web pages.

The user interface is placed in the middle of the screen because it was the fastest way to implement similar to the `SearchBar` as seen in Figure 5.1. Assuming using a low resolution screen a placement on the side could be of advantage because it takes up less space in the center where normally the focus of the users attention is. In this version so far the user can close the `TimeSeriesBar` and the animation will continue in the background.

With introducing new standards for data sets handled in Escher it got more complicated getting it in the right format. The web interface could provide an import tool in form of a plug-in to carry out the data preparation directly. The user could specify which data set to display on which point in time and perform clean-up like normalization. Since Escher offers an export function

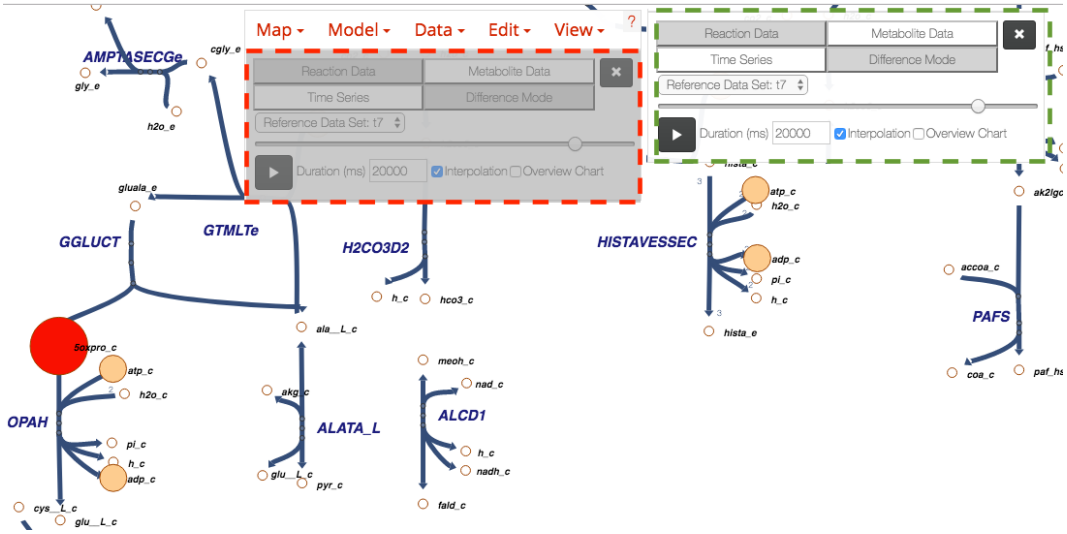


Figure 5.1: Possible improvement of graphical user interface. On screens with low resolution the centered `TimeSeriesBar` is not ideal because it takes up a lot of space in the center of the screen where normally the focus of the users attention is. Besides the possibility to close the window while it plays the animation it can be of interest to control the time series while playing. So a position on the side of the Menu would be an improvement.

for maps, a way to record and save the time series directly can be considered.

5.2 Visualization

An example how to use Escher to visualize a time series can be found in the application example Section 3.2. However, with further modifications a more straightforward way to visualize relative and absolute changes in concentration would be changing the appearance to a "filling level" model as shown in Figure 5.2. In this kind of visualization, the shape of the metabolites are changed from a circle to a rectangular shape and are filled corresponding to the absolute concentration of the metabolite. No filling shows the minimum while a full level the maximum value. The color tone of the metabolite object indicates its value relative to all other values in the network. This way the user can spot changes in concentration even if the relative value is low, while at the same time gaining an overview of the metabolite concentrations in comparison to all others. Also, there is no more overlapping with other metabolite circles if the radius is too big. Changes could be applied to indicate reaction flow speed as flowing waves to easier see the direction of a reaction.

Metabolic pathways can lead through different parts of a cell since all occurring reactions take place in specific compartments [Mar10]. In the SBML standard and are part of the map these compartments can already be included

[GRSW06]. When it comes to understanding which reaction is localized where in the cell compartments can be visualized as surrounding boxes.

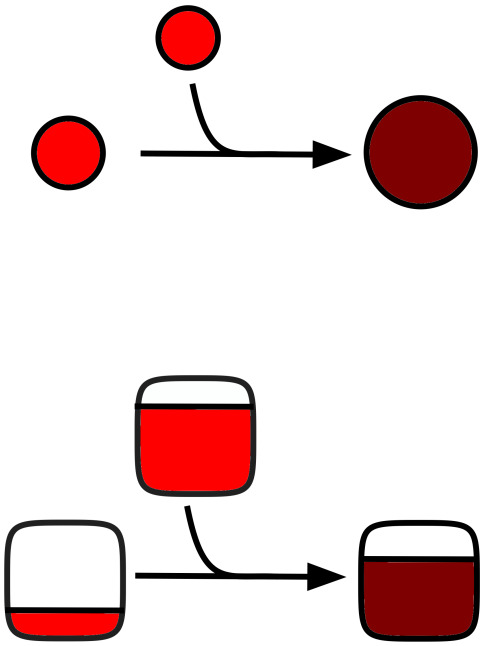


Figure 5.2: Another possible improvement of the visualization techniques in Escher is the "filling level" model. The shape of the metabolites are changed from a circle to a rectangular shape and are filled corresponding to the absolute concentration of the metabolite. The color of the metabolite object indicates its value relative to all other values in the network. While the upper node is small in the upper illustration in the lower illustration it becomes clear that in this state it has reached a high concentration over the course of time depending on its maximum value.

Chapter 6

Conclusion and Outlook

All the modifications made to Escher in this thesis were essential to enable a time series data visualization. New standards for loading data in Escher were created so the user is able to name the data sets according to time points or conditions. The code base was extended to accept, check and process this data. Creating a user interface for controlling the new features was also part of the work. A considerable part of the achievements also lay under the surface on code compatibility and stability.

It is now possible to show a time series or difference comparison of data sets in an animation. Interpolation techniques offer a better understanding of the course of the data series and give a smoother playback. To give an insight of the features several application examples are provided in this thesis.

In conclusion, with the accomplishments made in this thesis Escher can be used to answer specific questions in the context of metabolism additionally through visualizing time series data.

Besides time series visualization Escher has even more potential. Only if a metabolic pathway has a clear and understandable layout a meaningful visualization can be accomplished and results generated. This is also elementary for a time series data visualization. Layout algorithms are still in development and a lot of shapes must be done by hand. Until further automation is possible, Escher could be extended with tools to manually apply geometric shapes on a group of nodes. Its nature as an open source tool and with community feedback this program can be modified in a lot of sensible ways to make it even more useful in the future.

Appendix A

Further Materials and CD Content

- Thesis in PDF format
- Source Code
  - Escher Time Series Data Visualization source code
  - Escher Test code
- Example data
  - Data standards mock-ups
  - PLT map and metabolite data
  - R script for data preparation
- Application Example Videos
  - Demonstration of Escher in Time Series mode with PLT data
  - Time series animation of TCA cycle
  - Time series animation with overview chart
  - Difference Mode with and without sliding window

Bibliography

[AKK+09] Mario Albrecht, Andreas Kerren, Karsten Klein, Oliver Kohlbacher, Petra Mutzel, Wolfgang Paul, Falk Schreiber, and Michael Wybrow. On open problems in biological network visualization. In *International Symposium on Graph Drawing*, pages 256–267. Springer, 2009.

[BBDW14] Fabian Beck, Michael Burch, Stephan Diehl, and Daniel Weiskopf. The state of the art in visualizing dynamic graphs. *EuroVis STAR*, 2, 2014.

[BMKP14] Aarash Bordbar, Jonathan M. Monk, Zachary A. King, and Bernhard O. Palsson. Constraint-based models predict metabolic and associated cellular functions. *Nat Rev Genet*, 15(2):107–120, 02 2014.

[BOH11] M. Bostock, V. Ogievetsky, and J. Heer. D3; data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, Dec 2011.

[BT13] Sam Behjati and Patrick S Tarpey. What is next generation sequencing? *Archives of Disease in Childhood-Education and Practice*, 98(6):236–238, 2013.

[DNW13] Peter Droste, Katharina Nöh, and Wolfgang Wiechert. Omix–a visualization tool for metabolic networks with highest usability and customizability in focus. *Chemie Ingenieur Technik*, 85(6):849–862, 2013.

[DP14] Andreas Dräger and Bernhard Ø Palsson. Improving collaboration by standardization efforts in systems biology. *Frontiers in bioengineering and biotechnology*, 2, 2014.

[FMKT03] Akira Funahashi, Mineo Morohashi, Hiroaki Kitano, and Naoki Tanimura. Celldesigner: a process diagram editor for gene-regulatory and biochemical networks. *Biosilico*, 1(5):159–162, 2003.

[GRS+15] Ralph Gauges, Ursula Rost, Sven Sahle, Katja Wengler, and Frank T Bergmann. The systems biology markup language (sbml) level 3 package: Layout, version 1 core. *Journal of Integrative Bioinformatics (JIB)*, 12(2):550–602, 2015.

[GRSW06] Ralph Gauges, Ursula Rost, Sven Sahle, and Katja Wegner. A model diagram layout extension for sbml. *Bioinformatics*, 22(15):1879–1885, 2006.

[GW12] Nils Gehlenborg and Bang Wong. Points of view: Networks. *Nat Meth*, 9(2):115–115, 02 2012.

[HFS+03] Michael Hucka, Andrew Finney, Herbert M Sauro, Hamid Bolouri, John C Doyle, Hiroaki Kitano, Adam P Arkin, Benjamin J Bornstein, Dennis Bray, Athel Cornish-Bowden, et al. The systems biology markup language (sbml): a medium for representation and exchange of biochemical network models. *Bioinformatics*, 19(4):524–531, 2003.

[HK11] Richard P Horgan and Louise C Kenny. 'omic'technologies: genomics, transcriptomics, proteomics and metabolomics. *The Obstetrician & Gynaecologist*, 13(3):189–195, 2011.

[HK16] Barbara J Hunnicutt and Martin Krzywinski. Points of view: Pathways. *Nat Meth*, 13(1):5–5, 01 2016.

[KDE+15] Zachary A King, Andreas Dräger, Ali Ebrahim, Nikolaus Sonnenschein, Nathan E Lewis, and Bernhard O Palsson. Escher: a web application for building, sharing, and embedding data-rich visualizations of biological pathways. *PLoS computational biology*, 11(8):e1004321, 2015.

[Kit02] +Hiroaki" Kitano. Systems biology: A brief overview. *Science*, 295(5560):1662–1664, 2002.

[Mar10] William Martin. Evolutionary origins of metabolic compartmentalization in eukaryotes. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 365(1541):847–855, 2010.

[MPH09] Monica L Mo, Bernhard Ø Palsson, and Markus J Herrgård. Connecting extracellular metabolomic measurements to intracellular flux states in yeast. *BMC systems biology*, 3(1):37, 2009.

[OCN+11] Jeffrey D Orth, Tom M Conrad, Jessica Na, Joshua A Lerman, Hojung Nam, Adam M Feist, and Bernhard Ø Palsson. A comprehensive genome-scale reconstruction of escherichia coli metabolism—2011. *Molecular systems biology*, 7(1):535, 2011.

[OFP10] Jeffrey D Orth, Ronan MT Fleming, and Bernhard O Palsson. Reconstruction and use of microbial metabolic networks: the core escherichia coli metabolic model as an educational guide. *EcoSal plus*, 2010.

[OTP10] Jeffrey D Orth, Ines Thiele, and Bernhard O Palsson. What is flux balance analysis? *Nat Biotech*, 28(3):245–248, 03 2010.

[PSR+14] Giuseppe Paglia, Ólafur E Sigurjónsson, Óttar Rolfsson, Soley Valgeirsdottir, Morten Bagge Hansen, Sigurður Brynjólfsson, Sveinn Gudmundsson, and Bernhard O Palsson. Comprehensive metabolomic study of platelets reveals the expression of discrete metabolic phenotypes during storage. *Transfusion*, 54(11):2911–2923, 2014.

[SMO+03] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S Baliga, Jonathan T Wang, Daniel Ramage, Nada Amin, Benno Schwikowski, and Trey Ideker. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome research*, 13(11):2498–2504, 2003.

[SPCP10] Jan Schellenberger, Junyoung O Park, Tom M Conrad, and Bernhard Ø Palsson. Bigg: a biochemical genetic and genomic knowledgebase of large scale metabolic reconstructions. *BMC bioinformatics*, 11(1):213, 2010.

[Tea14] R Core Team. R: A language and environment for statistical computing. vienna, austria: R foundation for statistical computing; 2014, 2014.

[TRB+14] Alex Thomas, Sorena Rahmanian, Aarash Bordbar, Bernhard Ø Palsson, and Neema Jamshidi. Network reconstruction of platelet metabolism identifies metabolic signature for aspirin resistance. *Scientific reports*, 4, 2014.

[VHK+12] Corinna Vehlow, Jan Hasenauer, Andrei Kramer, Julian Heinrich, Nicole Radde, Frank Allgöwer, and Daniel Weiskopf. Uncertainty-aware visual analysis of biochemical reaction networks. In *Biological Data Visualization (BioVis), 2012 IEEE Symposium on*, pages 91–98. IEEE, 2012.

[XZAY11] Momiao Xiong, Zhongming Zhao, Jonathan Arnold, and Fuli Yu. Next-generation sequencing. *Journal of BioMed Research*, 2010, 2011.

Selbständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Bachelorarbeit selbständig und nur mit den angegebenen Hilfsmitteln angefertigt habe und dass alle Stellen, die dem Wortlaut oder dem Sinne nach anderen Werken entnommen sind, durch Angaben von Quellen als Entlehnung kenntlich gemacht worden sind. Diese Bachelorarbeit wurde in gleicher oder ähnlicher Form in keinem anderen Studiengang als Prüfungsleistung vorgelegt.

Ort, Datum Unterschrift