Quantitative Biology Center (QBiC)

Dr. Sven Nahnsen
Andreas Friedrich, Sven Fillinger

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Summer Semester 2017

# Data Management in Quantitative Biology

(BIOINF4399C)

## Project 4: Integrated Workflow for automated Transcription Start Site Prediction in Prokaryotes

Hand-in date: 06.07.2017

Presentation date: 13.07.2017

**Guidelines:**

- Please send your solutions (code and pdf) in a zip file to `sven.fillinger@qbic.uni-tuebingen.de`.
- Your code should be well-documented. If needed, provide an explanation on how to run it.
- Send your GitHub account to your supervisor (if you don't have an account, create one).
- Your project repository is `https://github.com/qbicsoftware/2017_workflow_project.git`
- Commit your code to GitHub and open `Issues` for discussion.
- Start early, so eventual problems can be fixed in time.

**General Information**

During this project, you will create an own bioinformatic workflow with one of three emerging workflow languages in scientific research and Docker. This project is experimental and challenges are expected to occur. However, succeeding parts of the project will be transported into a real production workflow system at QBiC. I will particular look on how you approach upcoming challenges and how you evaluate your choices. It will be a good preparation for your time after the university, as tasks like this might face you as a bioinformatian/research scientist at your future job position very likely.

**Task 1** *Evaluation of workflow languages (Research part)*

In the last couple of years, a lot of workflow languages (WLs) have emerged from the science community. Three prominent 'newcomers' are:

- **Snakemake** (`https://snakemake.readthedocs.io/en/stable/`)
- **Nextflow** (`https://www.nextflow.io/`)
- **WDL** (`https://software.broadinstitute.org/wdl/`)

The first task is to evaluate the three WLs and compare them. Collect basic information like licensing, source code availability, online community and more. Further information should contain installability, underlying programming language, projects/companies working with either of these WLs.

In addition, inform yourself about information like Docker support, Cluster/Cloud/Grid integration, job scheduler support (look for executors, and if they are able to talk to an underlying scheduler system like MOAB).

If you had to decide for one of these WLs, which one would you select, explain why (also if you are unsure about a decision, write why). Which criteria determine your decision?

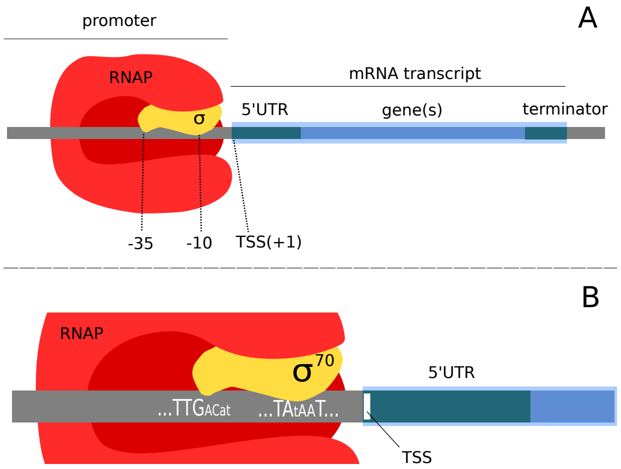

Abbildung 1: Schematic overview of the structural elements that play a role during gene expression in prokaryotes. **A**: The RNA polymerase (RNAP) is mediated to the promoter by an additionally bound $\sigma$-factor, which recognizes areas on the DNA around -35 and -10 nucleotide away from the real transcription start site (TSS). The transcription starts at +1 (TSS) and is part of the resulting mRNA transcript, which consists of a 5' untranslated region (UTR), one or more genes and a terminator sequence. **B**: The sequences around -35 and -10 are conserved and specific for different $\sigma$-factors. The example shows the sequences for the $\sigma^{70}$-factor in *E. coli*. Letter sizes indicate conservation level of single nucleotides

**Task 2** *Implementation of a TSS prediction workflow (Practical part)*
Now that you have (hopefully) decided on a WL that you want to test, this part involves the implementation of a workflow of a transcription start site (TSS) prediction pipeline.

### TSS prediction

You will get a more detailed description on transcription start sites and their importance in gene regulation and thus their relevance in scientific research. For now let us focus on some important facts.
A TSS is by definition the exact position at base level, where the DNA is transcribed into mRNA. Have a look at Figure 1, which shows a schematic mechanism overview of gene transcription in prokaryotes. The RNA polymerase (red) is mediated to the promoter region by binding sigma factors ($\sigma$). The sigma factor not only enables binding but also specificity on the sequence motif in the promoter region. There are several sigma factors in prokaryotes, and as you might probably already have guessed, they regulate different sets of genes. So they are an important part in the whole gene regulatory network.
In order to characterize promoter regions for i.e. mutations, motif analysis and subsequent regulatory elements such as riboswitches, the exact determination of the TSS is of great importance. Several tools emerged during the last couple of years, that provide automated TSS prediction. One of them is TSSpredator[2], which is among TSSar[1] and TSSer the most feature rich TSS prediction tool. In this course, we want to focus on TSSpredator, as QBiC and Prof. Dr. Kay Nieselt[1] are currently planning to integrate it in a complete TSS prediction workflow.
So what you are contributing to this project might actually go to an online production workflow!

### The workflow

Let's have a look at the conceptional scheme of the TSS prediction workflow (Figure 2). You will immediately recognize, that you need to integrate file preprocessing and conversion steps. These workflow steps or modules are symbolized with a gear icon. Workflow modules are exchangeable. You can use different software performing the same task, or different versions of the same software. This gives your workflow high flexibility and enables quick customization without rewriting any new code, once you brought several modules to life.
For this course, you are implementing TSSpredator and some of its[2] utility tools (tsstools, TSSarWrapper).

[1] Integrative Transcriptomics department
[2] depending on the development progress of TSSpredator and yours

What the different tools are doing exactly, will be discussed in more detail, once you have decided for this project with your supervisor.

**Workflow modules with Docker**

As part of this project, you should implement all workflow modules as Docker[3] containers. Please read up a little bit for yourself on this topic, as it will not be covered until later lecture slides in the DMQB lecture. Please use a robust Linux distribution as base operation system within the Docker container (like Centos 7).
I suggest that you start building your workflow with a strategy similar like this:

1. Install Docker on a UNIX based operation system (preferably you use notebook with a Linux distro).
2. Get a basic Docker image with Centos 7 (`https://hub.docker.com/_/centos/`).
3. Start the container and run an interactive shell within the container and execute `yum -y update` in order to update the OS.
4. Learn how to mount volumes from the host OS into the Docker container (so your software running inside your container can access data from outside the container).
5. Install a short-read sequence read mapping software of your choice in the container.
6. Run the container and execute the mapper. Also mount a volume containing the proper input files, so that the mapper will find them.

Once you have figured out the way how to start a Docker container and process input files with it and access the output files, you can dockerize all modules of the workflow in the same way, testing each module (here always encapsulated by an own docker container) independently.

**Task 3** *Your first complete workflow*

Once every module/container works properly for itself, you can start implementing a workflow in the WL of your choice and figure out, how output from one workflow module is passed to the next module. Last but not least create an additional mapper container, that runs a different mapping software.
Comment on the changes you have to make in your workflow to exchange the mapping module. What impact does it have on the final TSS prediction output? How could such a workflow contribute to reproducible and transparent research? Could Docker Hub be a part of it? Please justify your answer.

# Literatur

[1] Fabian Amman, Michael T. Wolfinger, Ronny Lorenz, Ivo L. Hofacker, Peter F. Stadler, and Sven Findeiß. TSSAR: TSS annotation regime for dRNA-seq data. *BMC Bioinformatics*, 15:89.

[2] Gaurav Dugar, Alexander Herbig, Konrad U. Förstner, Nadja Heidrich, Richard Reinhardt, Kay Nieselt, and Cynthia M. Sharma. High-resolution transcriptome maps reveal strain-specific regulatory features of multiple campylobacter jejuni isolates. *PLOS Genet*, 9(5):e1003495.

[3] `https://www.docker.com/what-docker`

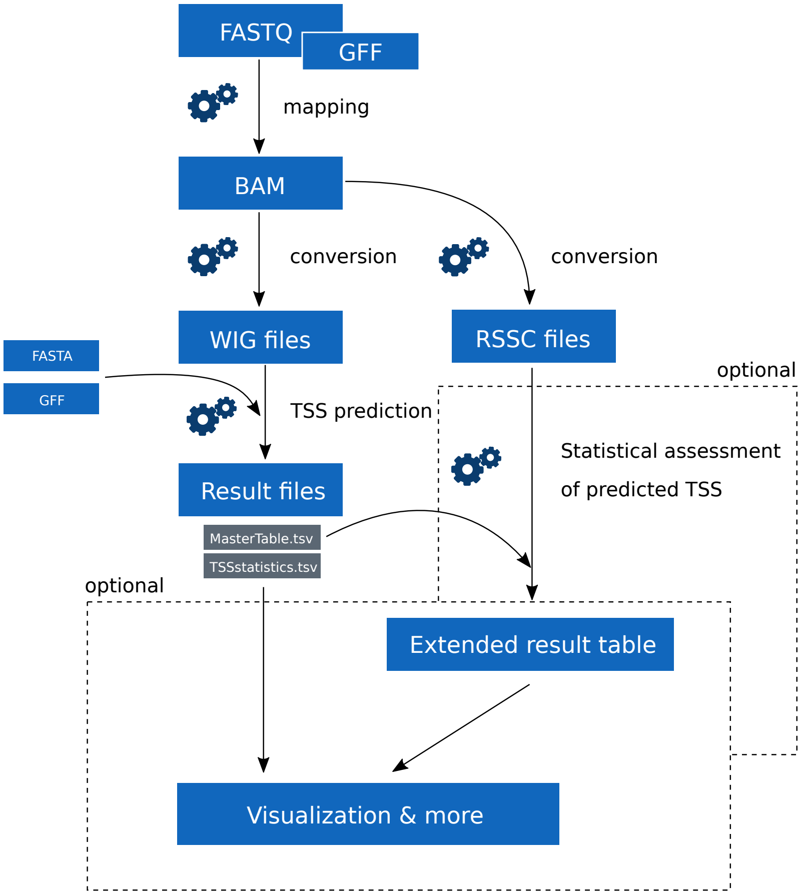

Abbildung 2: Conceptional scheme of a complete TSS prediction worklflow. Every gear symbol represents an own workflow step. The preceding and subsequent data types give you important information about the input and output elements of each step. Every step is performed by one software tool. A list of the necessary tools can be found in the task description. **Optional:** The workflow steps framed in dashed boxes are optional, let's see how far you get.