# Digestiflow: from BCL to FASTQ with ease

Manuel Holtgrewe[1,2], Mikko Nieminen[1,3], Clemens Messerschmidt[1,2], Dieter Beule[1,3]

[1] Berlin Institute of Health, Core Unit Bioinformatics, Charitéplatz 1, 10117 Berlin

[2] Charité – Universitätsmedizin Berlin, Charitéplatz 1, 10117 Berlin

[3] Max-Delbrück-Center for Molecular Medicine, Robert-Rössle-Straße 10, 13125 Berlin

Corresponding Author:
Dieter Beule[1,3]
Charitéplatz 1, 10117 Berlin
Email address: dieter.beule@bihealth.de

## Abstract

Management raw sequencing data and its preprocessing (conversion into sequences and demultiplexing) remains a challenging topic for groups running sequencing devices. They face many challenges in such efforts and solutions ranging from manual management of spreadsheets to very complex and customized LIMS systems handling much more than just sequencing raw data. In this manuscript, we describe the software package DigestiFlow that focuses on the management of Illumina flow cell sample sheets and raw data. It allows for automated extraction of information from flow cell data and management of sample sheets. Furthermore, it allows for the automated and reproducible conversion of Illumina base calls to sequences and the demultiplexing thereof using bcl2fastq and Picard Tools, followed by quality control report generation.

## 1 Introduction

Laboratories operating modern sequencing facilities face a multitude of challenges. These include sample tracking, raw data preprocessing (conversion of raw sequencer output into sequences and demultiplexing of pooled experiments which is usually done in the same step), quality control of sequencing results, and delivery to the requesting party. While there is no clear consensus of what comprises a Laboratory Information Management System (LIMS), the term LIMS is often used to describe systems supporting these step. Simple “pure peopleware” implementations consist of spreadsheets on network shares while comprehensive commercial packages such as Illumina BaseSpace Clarity LIMS offer highly adjustable but very expensive

solutions. A number of academic and open solutions fall in between, offering a variable number of features and degrees of customizability.

The general lack of agreement of what a LIMS should cover or not cover stems from the fact that sequencing laboratories alone differ greatly. Areas of difference include the types of samples accepted (tissues/blood, DNA/RNA, final libraries/pools, or a subset thereof), and the type of data generated (raw base calls, sequences, aligned reads, or bioinformatics analytical reports). In addition, the surrounding information technology (IT) infrastructure varies greatly as does the degree of integration with such additional IT systems.

In this manuscript we present our approach DigestiFlow (DF) that addresses the different needs of organizations by focusing on a small, well-defined subset of tasks: management of Illumina flow cell and sample sheet information and orchestrating the step converting base calls to sequences and demultiplexing pooled sequencing runs. To the best knowledge of the authors, in this domain DF offers unparalleled functionality. Flow cells can be filled with an arbitrary combination of libraries using any combination of index and molecular barcode reads. DF also supports the barcode being part of the template sequence. DF provides extensive features for sanity checking and comparison of expected indexing reads with those actually seen in the raw base call data.

This is particularly important in an era where technologies such as single cell and low input sequencing require an ever-growing complexity of barcoding and indexing schemes and the amount of sequencer throughput is growing dramatically. We have encountered flow cells with more than 600 libraries and expect this to grow with increasing sequencer throughput.

A fundamental link to central IT is the integration with existing authentication infrastructure via directory servers, e.g., Microsoft ActiveDirectory (AD). DF supports linking accounts to central AD instances as well as using user accounts that only exist within the system. Beyond this, the system provides its functionality through a REST API (representational state transfer application programmable interface application programming interface) such that other services can be easily integrated. Instead of covering all possible functionality and sample tracking schemes, DF avoids the complexity of a monolithic system and can be integrated as a part of a modular system. However, it can also just be standalone without integration with any other system.

## 2 Methods

DigestiFlow (DF) consist of three major components. The architecture of the system is shown in Figure 1. The figure also shows interaction with a minimal set of external systems.

### 2.1 Digestiflow Server

DF Server is a web app implemented with the SODAR-Core (Nieminen, Stolpe, Schumann, Holtgrewe, & Beule, n.d.) and Django frameworks in the Python programming language. It uses a PostgreSQL database system. It allows for the curation of flow cells and libraries together with arbitrarily complex index and barcoding schemes. Barcodes can be organized in barcode sets such that their sequence can be entered once and subsequently be referred to by name. Furthermore, sequencing machines can be registered with their main properties (e.g., whether the second barcode read needs to be reverse-complemented, depending on the paired indexing workflow used). DF Server allows the visualization of barcodes detected by DF Client in the BCL files (see Figure 2) and compares them to the libraries and barcodes entered by the users into the flow cell sample sheet. DF Server provides a number of sanity checks for both barcodes from raw sequencing data, including a barcode frequency distribution and recognizing expected spike-ins such as PhiX sequence. It can also cross check between sample sheets and barcodes in the raw sequences, detecting barcodes present in one but missing in the other. Furthermore, users can add comments to flow cells and attach arbitrary files, which is useful for exchanging spreadsheets or concentration measurement reports from the wet lab.

### 2.2 DF Client

DF Client is meant to be called periodically via a cron job to monitor the storage volume where sequencer(s) write output data. It reads the metadata files and registers any new flow cell with DF Server (or alternately, flow cells can be pre-registered in DF Server and their properties are then updated by DF Client). Once the barcodes have been sequenced completely, the DF Client extracts and evaluates their sequences and posts this information to DF Server. The client also detects when sequencing has succeeded (and various failure conditions) and updates the information in the server. DF Client is written in the Rust programming language.

### 2.3 Digestiflow Demux

DF Demux is also meant to watch the storage volume where the sequencers write their output data. Once sequencing of a flow cell is complete and marked as ready in DF Server, it starts the preprocessing by first obtaining the flow cell information from DF Server. Flow cells can be marked for delivery as base call (BCL) files, (possibly) demultiplexed sequences, or both. If raw BCL files are to be delivered, DF Server simply creates a TAR (tape archive) file for each lane that contains all the information required for demultiplexing this one lane.

For preprocessing, it first checks whether the flow cell can be processed by simply calling the Illumina vendor software bcl2fastq (version 1 or 2, depending on the needs of the raw data) and calls the program accordingly. Otherwise, it generates a series of calls to bcl2fastq and Picard

Tools (http://broadinstitute.github.io/picard/) to perform the required preprocessing. An example for this is the Agilent XT protocol where molecular barcode sequences are stored in the second barcode read which bcl2fastq does not support. We refer to homogenous flow cell loads that can be processed with the bcl2fastq as basic preprocessing while flexible preprocessing allows arbitrary combination of library indexing and barcoding schemes.

Once preprocessing is complete, FastQC (https://www.bioinformatics.babraham.ac.uk/projects/fastqc/) is run on the results and the quality control results are be collected with MultiQC (Ewels, Magnusson, Lundin, & Käller, 2016). Finally, the MultiQC report is posted as a message to the flow cell in DF Server using the REST API together with the log files as attachment. This then allows the sequencing staff to review the results and react accordingly. DF Demux is implemented in the Python programming language with a Snakemake (Köster & Rahmann, 2012) workflow using Bioconda (Grüning et al., 2018) for software package management.

# 3 Results

## 3.1 The States of a Flow Cell

DF tracks three components of sequencing: (a) the sequencing process itself, (b) preprocessing, and (c) data delivery. The possible state values differ for each of these three steps. See the manual in the Supplemental Material for full details, but they can be summarized as follows. Each step starts in the initial state. For preprocessing, an operator has to set the state of the preprocessing step to ready which signals DF Demux to start. Once the client detects that the sequencing of a flow cell has started the sequencing state changes to running. Similarly, once DF Demux has started, the preprocessing state changes to running. If sequencing or demultiplexing fails or succeeds, the corresponding state is updated accordingly (failed/success). For both, a human operator can set a special confirmed failure/success state manually. For example, a confirmed failure state will be set manually after they determine preprocessing failed due to overall low sequencing quality and it is not possible to “rescue” preprocesisng by fixing a sample sheet. Or a confirmed success state may be set after a human operator determines that QC passes visual inspection. A special success with warning state allows users to flag situations such as sequencing which succeeded for all but one lane due to technical issues.

For delivery, a human user has to set the state explicitly. This built-in system for keeping track of the delivery state is particularly useful if more than one user is handling data delivery, especially when used in conjunction with the message feature for leaving notes on flow cells.

## 3.2 Comparison with Existing Methods

Existing methods include the following: openBIS ELN-LIMS (Barillari et al., 2016) which builds on top of openBIS (Bauch et al., 2011) and has a high number of features for sample submission and -tracking yet also has a large number of dependencies, MendeLIMS (Grimes & Ji, 2014) which has basic sample tracking functionality yet is bound to a rigid data processing workflow, MISO (Masella et al., 2019) which offers basic sample tracking functionality yet does not include features for preprocessing, and Parkour LIMS (Anatskiy et al., 2019) which provides extensive sample tracking and advanced lab notebook features yet also does not integrate automated preprocessing. While being out of scope of this manuscript, we note that DF could be integrated with other software packages as long as they provide an API with additional code. The integration with Parkour LIMS appears particularly appealing as it is based on the same technology as DF (Python/Django) and has few other dependencies itself. Table 1 contains a comparison of the listed tools given some important features

### 3.3 Features for Improving Sequencing Results

**Sample Sheet Validation.** Based on practical experience, we greatly appreciate the automated comparison of observed adapter sequence content and sample sheet. Unexpected sequence in either set is an indication for possible errors. DF Server provides fine-grained control to acknowledge and suppress inconsistency warnings (after either fixing errors or accepting errors and then excluding corresponding data). Furthermore, common artifacts such as PhiX sequence are automatically recognized and show up as information rather than warnings or errors. Figure 2 shows an example.

**Reproducibility, Automation, and Quality Control.** The Digestiflow Client and Demux components are available from Bioconda as Conda packages and Docker images, thus allowing for future proof installations and creating reproducible workflows. By offering REST APIs and two useful client applications, DF greatly supports sequencing and demultiplexing operators in automating their work. Further automation can be added later as the APIs are open. Automated quality control using FastQC and aggregation using MultiQC also allows users to spot problems earlier (together with the sample sheet adapter checks described above). In our experience this allows for the early detection of many common issues. For example, from time to time, it occurs that the same adapter was used for two different libraries in the same lane. This error might be hard to spot on paper or in spreadsheets but applications such as DF Server can easily detect and report such problems similar to the example shown in Figure 2.

## References

Anatskiy, E., Ryan, D. P., Grüning, B. A., Arrigoni, L., Manke, T., & Bönisch, U. (2019). Parkour LIMS: high-quality sample preparation in next generation sequencing. Bioinformatics. doi:10.1093/bioinformatics/bty820

Barillari, C., Ottoz, D. S. M., Fuentes-Serna, J. M., Ramakrishnan, C., Rinn, B., & Rudolf, F. (2016). openBIS ELN-LIMS: an open-source database for academic laboratories. Bioinformatics, 32(4), 638–640. doi:10.1093/bioinformatics/btv606

Bauch, A., Adamczyk, I., Buczek, P., Elmer, F.-J., Enimanev, K., Glyzewski, P., … Rinn, B. (2011). openBIS: a flexible framework for managing and analyzing complex data in biology research. BMC Bioinformatics, 12(1), 468. doi:10.1186/1471-2105-12-468

Ewels, P., Magnusson, M., Lundin, S., & Käller, M. (2016). MultiQC: summarize analysis results for multiple tools and samples in a single report. Bioinformatics, 32(19), 3047–3048. https://doi.org/10.1093/bioinformatics/btw354

Grimes, S. M., & Ji, H. P. (2014). MendeLIMS: A web-based laboratory information management system for clinical genome sequencing. BMC Bioinformatics. doi:10.1186/1471-2105-15-290

Grüning, B., Dale, R., Sjödin, A., Chapman, B. A., Rowe, J., Tomkins-Tinch, C. H., … Köster, J. (2018). Bioconda: sustainable and comprehensive software distribution for the life sciences. Nature Methods, 15(7), 475–476. doi:10.1038/s41592-018-0046-7

Köster, J., & Rahmann, S. (2012). Snakemake--a scalable bioinformatics workflow engine. Bioinformatics, 28(19), 2520–2522. doi:10.1093/bioinformatics/bts480

Masella, A., Cooke, D., Armstrong, H., Davey, R., DeBat, T., Bian, X., … Leipzig, J. (2019). miso-lims/miso-lims: v0.2.183. doi:https://10.5281/ZENODO.3341739

Nieminen, M., Stolpe, O., Holtgrewe, M., Beule, D. SODAR Core: a Django-based framework for scientific data management and analysis web apps. under review

## Tables

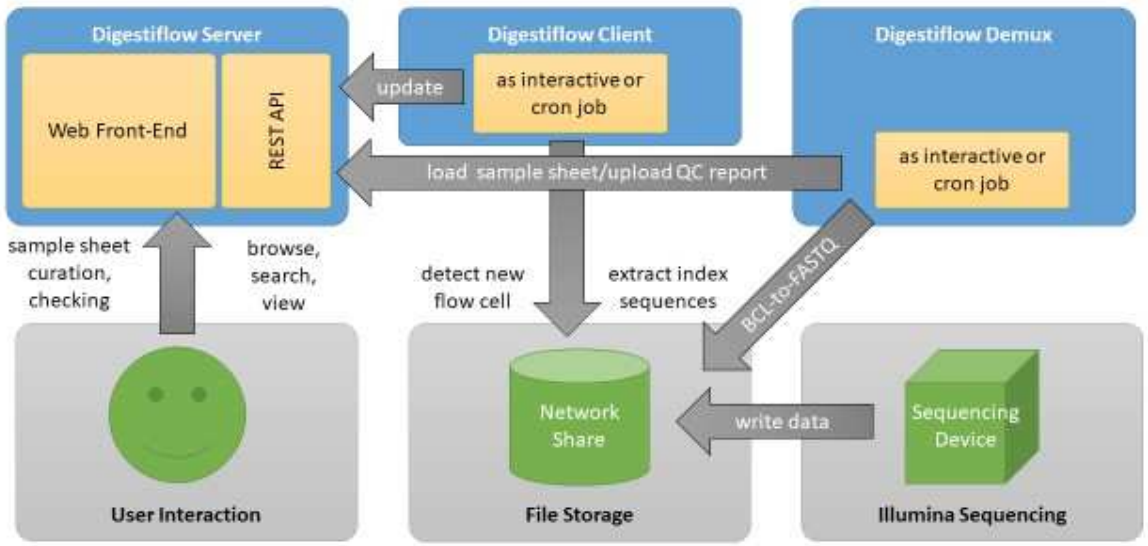
**Table 1** Comparison important properties and features in commercial and free software for the management of Illumina flow cells information popular in the sequencing community based.

| Metric | DF | BSCL | OBLE | ML | MISO | PL |
|---|---|---|---|---|---|---|
| License | MIT | com. | f.f.n. | f.f.n. | GPL | GPL |
| Self-Hosted | ✓ | – | ✓ | ✓ | ✓ | ✓ |
| LDAP Auth | ✓ | ✓ | ✓ | ✓ | ✓ | – |
| (REST) API | ✓ | ✓ | ✓ | – | ✓ | ✓ |
| Sample Tracking | min. | adv. | basic | basic | basic | adv. |
| Basic Preproc. | ✓ | ✓ | ✓ | ✓ | – | ✓ |
| Flexible Preproc. | ✓ | – | – | – | – | – |
| Sheet Checks | ✓ | – | – | – | – | – |
| BCL Checks | ✓ | – | – | – | – | – |

Abbreviations used in the table: DF (DigestiFlow), BSCL (BaseSpace Clarity LIMS), OBLE (OpenBIS LIMS-ELN), ML (MendeLIMS), PL (Parkour LIMS), com. (commercial), f.f.n. (free for non-commercial), min. (minimal), adv. (advanced), preproc. (preprocessing).

## Figures

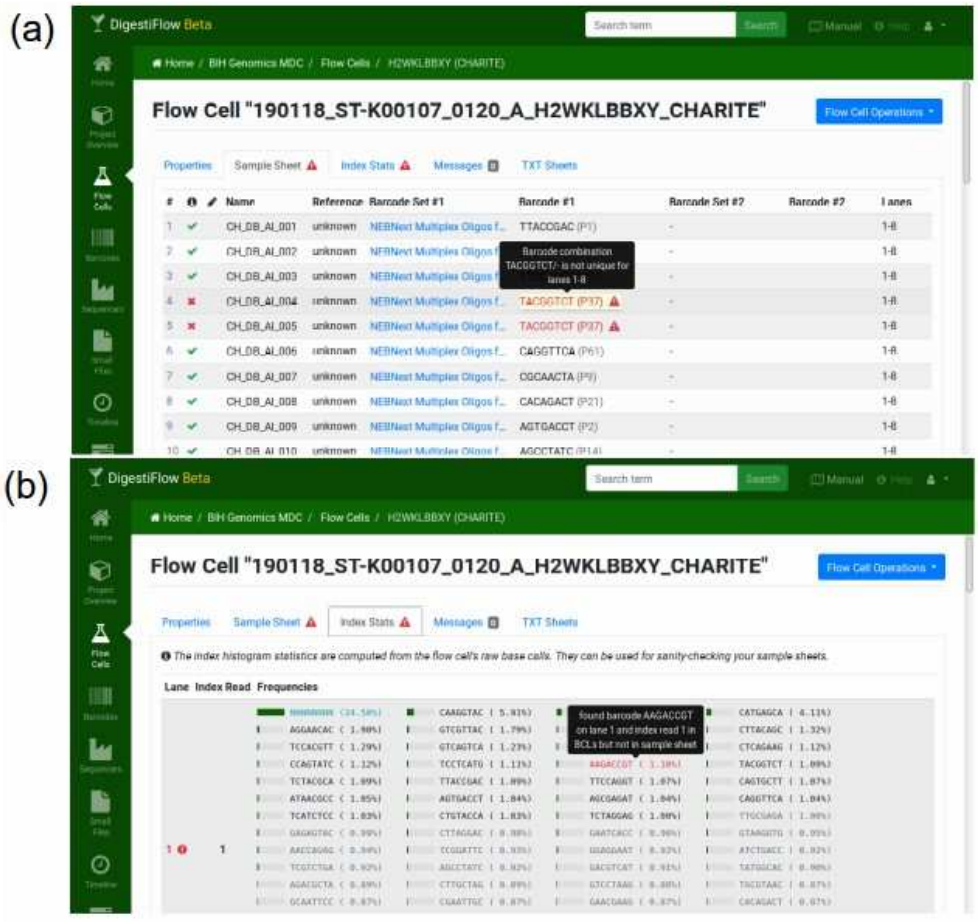
**Figure 1** Architectural overview. Sequencing instruments write data to a specified file system storage. A periodically running DF Client detects new flow cells and registers them with the DF Server. Once sequencing is complete and sample sheet information has been approved by the operator, DF Demux performs the conversion to FASTQ files and creates all QC reports. Users can not only browse and view, but also manage and curate flow cells and their sample sheets through DF Server.



219

220**Figure 2**

221When adding the sample sheet (not shown), the operator made a small mistake. The adapter P37
222is given twice for the same lane in the sample sheet while the adapter sequence “AAGACCGT”
223occurs in the raw base calls but not in the sample sheet. This information can then be used for
224debugging sample sheet information. This is highlighted in the sample sheet (a) and the display
225of the adapters read from the raw base call data (b).



## Supplemental Material

- Digestiflow Server Documentation