CUBI-SAK Documentation

Release 0.3.0+0.gf386523.dirty

Core Unit Bioinformatics

May 05, 2021

Installation Getting Started

1 Installation 3

2 Command Line Interface 5

3 Manual for `isa-tpl` 41

4 Manual for `isa-tab` 43

5 Manual for `ingest-fastq` 45

6 Manual for `sea-snap itransfer-results` 49

7 Manual for `sea-snap write-sample-info` 51

8 Use Case: Exomes 55

9 Use Case: Single Cell 61

10 Credits 65

11 HISTORY 67

12 License 69

Python Module Index 71

Index 73

Installation & Getting Started Instructions for the installation of the module and some examples to get you started.

Installation
API documentation

Manual This section contains manuals for specific commands.

Creating ISA-tab files
Annotating ISA-tab files
Upload raw data to SODAR
Upload raw data to SODAR
Create a sample info file for Sea-snap

Use cases Use cases for common processing tasks.

Exome sequencing
Clinical single cell pipeline

Project Info More information on the project, including the changelog, list of contributing authors, and contribution instructions.

Authors
History
License

CHAPTER 1

Installation

Prerequisites when using conda:

```
$ conda create -n cubi-tk python=3.7
$ conda activate cubi-tk
```

Clone CUBI-SAK and install:

```
$ git clone git@cubi-gitlab.bihealth.org:CUBI/Pipelines/cubi-tk.git
$ cd cubi-tk
$ pip install -e .
```

For building the manual or running tests you will need some more packages.

```
$ pip install -r requirements/develop.txt
```

1.1 Run tests

```
$ make test
```

1.2 Build manual

```
$ cd docs_manual
$ make clean html
```

CHAPTER 2

Command Line Interface

```
usage: cubi-tk [-h] [--verbose] [--version] [--config CONFIG]
               [--sodar-server-url SODAR_SERVER_URL]
               [--sodar-api-token SODAR_API_TOKEN]
               {isa-tpl,isa-tab,snappy,sodar,irods,org-raw,sea-snap} ...
```

2.1 Positional Arguments

cmd Possible choices: isa-tpl, isa-tab, snappy, sodar, irods, org-raw, sea-snap

2.2 Named Arguments

--verbose Increase verbosity.

Default: False

--version show program's version number and exit

2.3 Basic Configuration

--config Path to configuration file.

--sodar-server-url SODAR server URL key to use, defaults to env SODAR_SERVER_URL.

--sodar-api-token SODAR API token to use, defaults to env SODAR_API_TOKEN.

2.4 Sub-commands:

2.4.1 isa-tpl

Create of ISA-tab directories from predefined templates.

```
cubi-tk isa-tpl [-h]
                {single_cell_rnaseq,tumor_normal_dna,tumor_normal_triplets,germline,
↪generic,microarray,ms_meta_biocrates}
                ...
```

Positional Arguments

tpl Possible choices: single_cell_rnaseq, tumor_normal_dna, tumor_normal_triplets, germline, generic, microarray, ms_meta_biocrates

Sub-commands:

single_cell_rnaseq

When specifying the –var-* argument, you can use JSON syntax. Failing to parse JSON will keep the string value.

```
cubi-tk isa-tpl single_cell_rnaseq [-h]
                                   [--var-investigation-title VAR_INVESTIGATION_TITLE]
                                   [--var-sample-names VAR_SAMPLE_NAMES]
                                   [--var-a-measurement-type VAR_A_MEASUREMENT_TYPE]
                                   [--var-lib-kit VAR_LIB_KIT]
                                   [--var-batch VAR_BATCH]
                                   [--var-lib-kits VAR_LIB_KITS]
                                   [--var-instrument VAR_INSTRUMENT]
                                   [--var-center-name VAR_CENTER_NAME]
                                   [--var-center-contact VAR_CENTER_CONTACT]
                                   [--var-study-title VAR_STUDY_TITLE]
                                   [--var-i-dir-name VAR_I_DIR_NAME]
                                   [--var-s-file-name VAR_S_FILE_NAME]
                                   [--var-assay-prefix VAR_ASSAY_PREFIX]
                                   [--var-a-technology-type VAR_A_TECHNOLOGY_TYPE]
                                   [--var-a-measurement-abbreviation VAR_A_
↪MEASUREMENT_ABBREVIATION]
                                   [--var-assay-name VAR_ASSAY_NAME]
                                   [--var-sample-type VAR_SAMPLE_TYPE]
                                   [--var-lib-strategy VAR_LIB_STRATEGY]
                                   [--var-lib-selection VAR_LIB_SELECTION]
                                   [--var-lib-layout VAR_LIB_LAYOUT]
                                   [--var-lib-strand-specificity VAR_LIB_STRAND_
↪SPECIFICITY]
                                   [--var-library-name-mRNA VAR_LIBRARY_NAME_MRNA]
                                   [--var-library-name-sample-tag VAR_LIBRARY_NAME_
↪SAMPLE_TAG]
                                  output_dir
```

Positional Arguments

output_dir Path to output directory

Named Arguments

--var-investigation-title template variables 'investigation_title'

--var-sample-names template variables 'sample_names'

--var-a-measurement-type template variables 'a_measurement_type'

--var-lib-kit template variables 'lib_kit'

--var-batch template variables 'batch'

--var-lib-kits template variables 'lib_kits'

--var-instrument template variables 'instrument'

--var-center-name template variables 'center_name'

--var-center-contact template variables 'center_contact'

--var-study-title template variables 'study_title'

--var-i-dir-name template variables 'i_dir_name'

--var-s-file-name template variables 's_file_name'

--var-assay-prefix template variables 'assay_prefix'

--var-a-technology-type template variables 'a_technology_type'

--var-a-measurement-abbreviation template variables 'a_measurement_abbreviation'

--var-assay-name template variables 'assay_name'

--var-sample-type template variables 'sample_type'

--var-lib-strategy template variables 'lib_strategy'

--var-lib-selection template variables 'lib_selection'

--var-lib-layout template variables 'lib_layout'

--var-lib-strand-specificity template variables 'lib_strand_specificity'

--var-library-name-mRNA template variables 'library_name_mRNA'

--var-library-name-sample-tag template variables 'library_name_sample_tag'

tumor_normal_dna

When specifying the –var-* argument, you can use JSON syntax. Failing to parse JSON will keep the string value.

```
cubi-tk isa-tpl tumor_normal_dna [-h]
                                 [--var-investigation-title VAR_INVESTIGATION_TITLE]
                                 [--var-sample-names VAR_SAMPLE_NAMES]
                                 [--var-a-measurement-type VAR_A_MEASUREMENT_TYPE]
                                 [--var-lib-kit VAR_LIB_KIT]
                                 [--var-lib-kits VAR_LIB_KITS]
                                 [--var-instrument VAR_INSTRUMENT]
```

(continues on next page)

(continued from previous page)

```
                                    [--var-center-name VAR_CENTER_NAME]
                                    [--var-center-contact VAR_CENTER_CONTACT]
                                    [--var-study-title VAR_STUDY_TITLE]
                                    [--var-i-dir-name VAR_I_DIR_NAME]
                                    [--var-is-triplet VAR_IS_TRIPLET]
                                    [--var-s-file-name VAR_S_FILE_NAME]
                                    [--var-assay-prefix VAR_ASSAY_PREFIX]
                                    [--var-a-technology-type VAR_A_TECHNOLOGY_TYPE]
                                    [--var-a-measurement-abbreviation VAR_A_MEASUREMENT_
↪ABBREVIATION]
                                    [--var-assay-name VAR_ASSAY_NAME]
                                    [--var-sample-type VAR_SAMPLE_TYPE]
                                    [--var-lib-strategy VAR_LIB_STRATEGY]
                                    [--var-lib-selection VAR_LIB_SELECTION]
                                    [--var-lib-layout VAR_LIB_LAYOUT]
                                    output_dir
```

Positional Arguments

output_dir Path to output directory

Named Arguments

--var-investigation-title template variables ‘investigation_title’

--var-sample-names template variables ‘sample_names’

--var-a-measurement-type template variables ‘a_measurement_type’

--var-lib-kit template variables ‘lib_kit’

--var-lib-kits template variables ‘lib_kits’

--var-instrument template variables ‘instrument’

--var-center-name template variables ‘center_name’

--var-center-contact template variables ‘center_contact’

--var-study-title template variables ‘study_title’

--var-i-dir-name template variables ‘i_dir_name’

--var-is-triplet template variables ‘is_triplet’

--var-s-file-name template variables ‘s_file_name’

--var-assay-prefix template variables ‘assay_prefix’

--var-a-technology-type template variables ‘a_technology_type’

--var-a-measurement-abbreviation template variables ‘a_measurement_abbreviation’

--var-assay-name template variables ‘assay_name’

--var-sample-type template variables ‘sample_type’

--var-lib-strategy template variables ‘lib_strategy’

--var-lib-selection template variables ‘lib_selection’

--var-lib-layout template variables 'lib_layout'

tumor_normal_triplets

When specifying the –var-* argument, you can use JSON syntax. Failing to parse JSON will keep the string value.

```
cubi-tk isa-tpl tumor_normal_triplets [-h]
                                      [--var-investigation-title VAR_INVESTIGATION_
↪TITLE]
                                      [--var-sample-names VAR_SAMPLE_NAMES]
                                      [--var-a-measurement-type VAR_A_MEASUREMENT_
↪TYPE]
                                      [--var-lib-kit VAR_LIB_KIT]
                                      [--var-lib-kits VAR_LIB_KITS]
                                      [--var-instrument VAR_INSTRUMENT]
                                      [--var-center-name VAR_CENTER_NAME]
                                      [--var-center-contact VAR_CENTER_CONTACT]
                                      [--var-study-title VAR_STUDY_TITLE]
                                      [--var-i-dir-name VAR_I_DIR_NAME]
                                      [--var-is-triplet VAR_IS_TRIPLET]
                                      [--var-s-file-name VAR_S_FILE_NAME]
                                      [--var-assay-prefix VAR_ASSAY_PREFIX]
                                      [--var-a-technology-type VAR_A_TECHNOLOGY_TYPE]
                                      [--var-a-measurement-abbreviation VAR_A_
↪MEASUREMENT_ABBREVIATION]
                                      [--var-assay-name VAR_ASSAY_NAME]
                                      [--var-sample-type VAR_SAMPLE_TYPE]
                                      [--var-lib-strategy VAR_LIB_STRATEGY]
                                      [--var-lib-selection VAR_LIB_SELECTION]
                                      [--var-lib-layout VAR_LIB_LAYOUT]
                                     output_dir
```

Positional Arguments

output_dir Path to output directory

Named Arguments

--var-investigation-title template variables 'investigation_title'

--var-sample-names template variables 'sample_names'

--var-a-measurement-type template variables 'a_measurement_type'

--var-lib-kit template variables 'lib_kit'

--var-lib-kits template variables 'lib_kits'

--var-instrument template variables 'instrument'

--var-center-name template variables 'center_name'

--var-center-contact template variables 'center_contact'

--var-study-title template variables 'study_title'

--var-i-dir-name template variables 'i_dir_name'

--var-is-triplet template variables 'is_triplet'

--var-s-file-name template variables 's_file_name'

--var-assay-prefix template variables 'assay_prefix'

--var-a-technology-type template variables 'a_technology_type'

--var-a-measurement-abbreviation template variables 'a_measurement_abbreviation'

--var-assay-name template variables 'assay_name'

--var-sample-type template variables 'sample_type'

--var-lib-strategy template variables 'lib_strategy'

--var-lib-selection template variables 'lib_selection'

--var-lib-layout template variables 'lib_layout'

germline

When specifying the –var-* argument, you can use JSON syntax. Failing to parse JSON will keep the string value.

```
cubi-tk isa-tpl germline [-h]
                         [--var-investigation-title VAR_INVESTIGATION_TITLE]
                         [--var-sample-names VAR_SAMPLE_NAMES]
                         [--var-a-measurement-type VAR_A_MEASUREMENT_TYPE]
                         [--var-lib-kit VAR_LIB_KIT] [--var-batch VAR_BATCH]
                         [--var-lib-kits VAR_LIB_KITS]
                         [--var-instrument VAR_INSTRUMENT]
                         [--var-center-name VAR_CENTER_NAME]
                         [--var-center-contact VAR_CENTER_CONTACT]
                         [--var-study-title VAR_STUDY_TITLE]
                         [--var-i-dir-name VAR_I_DIR_NAME]
                         [--var-s-file-name VAR_S_FILE_NAME]
                         [--var-assay-prefix VAR_ASSAY_PREFIX]
                         [--var-a-technology-type VAR_A_TECHNOLOGY_TYPE]
                         [--var-a-measurement-abbreviation VAR_A_MEASUREMENT_
↪ABBREVIATION]
                         [--var-assay-name VAR_ASSAY_NAME]
                         [--var-sample-type VAR_SAMPLE_TYPE]
                         [--var-lib-strategy VAR_LIB_STRATEGY]
                         [--var-lib-selection VAR_LIB_SELECTION]
                         [--var-lib-layout VAR_LIB_LAYOUT]
                        output_dir
```

Positional Arguments

output_dir Path to output directory

Named Arguments

--var-investigation-title template variables 'investigation_title'

--var-sample-names template variables 'sample_names'

--var-a-measurement-type template variables 'a_measurement_type'

--var-lib-kit template variables 'lib_kit'

--var-batch template variables 'batch'

--var-lib-kits template variables 'lib_kits'

--var-instrument template variables 'instrument'

--var-center-name template variables 'center_name'

--var-center-contact template variables 'center_contact'

--var-study-title template variables 'study_title'

--var-i-dir-name template variables 'i_dir_name'

--var-s-file-name template variables 's_file_name'

--var-assay-prefix template variables 'assay_prefix'

--var-a-technology-type template variables 'a_technology_type'

--var-a-measurement-abbreviation template variables 'a_measurement_abbreviation'

--var-assay-name template variables 'assay_name'

--var-sample-type template variables 'sample_type'

--var-lib-strategy template variables 'lib_strategy'

--var-lib-selection template variables 'lib_selection'

--var-lib-layout template variables 'lib_layout'

generic

When specifying the –var-* argument, you can use JSON syntax. Failing to parse JSON will keep the string value.

```
cubi-tk isa-tpl generic [-h]
                        [--var-investigation-title VAR_INVESTIGATION_TITLE]
                        [--var-sample-names VAR_SAMPLE_NAMES]
                        [--var-a-measurement-type VAR_A_MEASUREMENT_TYPE]
                        [--var-lib-kit VAR_LIB_KIT]
                        [--var-organism VAR_ORGANISM] [--var-batch VAR_BATCH]
                        [--var-lib-kits VAR_LIB_KITS]
                        [--var-organisms VAR_ORGANISMS]
                        [--var-instrument VAR_INSTRUMENT]
                        [--var-center-name VAR_CENTER_NAME]
                        [--var-center-contact VAR_CENTER_CONTACT]
                        [--var-study-title VAR_STUDY_TITLE]
                        [--var-i-dir-name VAR_I_DIR_NAME]
                        [--var-s-file-name VAR_S_FILE_NAME]
                        [--var-assay-prefix VAR_ASSAY_PREFIX]
                        [--var-a-technology-type VAR_A_TECHNOLOGY_TYPE]
                        [--var-a-measurement-abbreviation VAR_A_MEASUREMENT_
↪ABBREVIATION]
                        [--var-assay-name VAR_ASSAY_NAME]
                        [--var-sample-type VAR_SAMPLE_TYPE]
                        [--var-lib-strategy VAR_LIB_STRATEGY]
                        [--var-lib-selection VAR_LIB_SELECTION]
                        [--var-lib-layout VAR_LIB_LAYOUT]
                        output_dir
```

Positional Arguments

output_dir Path to output directory

Named Arguments

--var-investigation-title template variables 'investigation_title'

--var-sample-names template variables 'sample_names'

--var-a-measurement-type template variables 'a_measurement_type'

--var-lib-kit template variables 'lib_kit'

--var-organism template variables 'organism'

--var-batch template variables 'batch'

--var-lib-kits template variables 'lib_kits'

--var-organisms template variables 'organisms'

--var-instrument template variables 'instrument'

--var-center-name template variables 'center_name'

--var-center-contact template variables 'center_contact'

--var-study-title template variables 'study_title'

--var-i-dir-name template variables 'i_dir_name'

--var-s-file-name template variables 's_file_name'

--var-assay-prefix template variables 'assay_prefix'

--var-a-technology-type template variables 'a_technology_type'

--var-a-measurement-abbreviation template variables 'a_measurement_abbreviation'

--var-assay-name template variables 'assay_name'

--var-sample-type template variables 'sample_type'

--var-lib-strategy template variables 'lib_strategy'

--var-lib-selection template variables 'lib_selection'

--var-lib-layout template variables 'lib_layout'

microarray

When specifying the –var-* argument, you can use JSON syntax. Failing to parse JSON will keep the string value.

```
cubi-tk isa-tpl microarray [-h]
                           [--var-investigation-title VAR_INVESTIGATION_TITLE]
                           [--var-sample-names VAR_SAMPLE_NAMES]
                           [--var-a-measurement-type VAR_A_MEASUREMENT_TYPE]
                           [--var-organism VAR_ORGANISM]
                           [--var-organisms VAR_ORGANISMS]
                           [--var-technology-platform VAR_TECHNOLOGY_PLATFORM]
                           [--var-array-design-ref VAR_ARRAY_DESIGN_REF]
```

(continues on next page)

(continued from previous page)

```
                                [--var-study-title VAR_STUDY_TITLE]
                                [--var-i-dir-name VAR_I_DIR_NAME]
                                [--var-s-file-name VAR_S_FILE_NAME]
                                [--var-assay-prefix VAR_ASSAY_PREFIX]
                                [--var-a-technology-type VAR_A_TECHNOLOGY_TYPE]
                                [--var-assay-name VAR_ASSAY_NAME]
                                [--var-terms VAR_TERMS]
                                output_dir
```

Positional Arguments

output_dir Path to output directory

Named Arguments

--var-investigation-title template variables 'investigation_title'

--var-sample-names template variables 'sample_names'

--var-a-measurement-type template variables 'a_measurement_type'

--var-organism template variables 'organism'

--var-organisms template variables 'organisms'

--var-technology-platform template variables 'technology_platform'

--var-array-design-ref template variables 'array_design_ref'

--var-study-title template variables 'study_title'

--var-i-dir-name template variables 'i_dir_name'

--var-s-file-name template variables 's_file_name'

--var-assay-prefix template variables 'assay_prefix'

--var-a-technology-type template variables 'a_technology_type'

--var-assay-name template variables 'assay_name'

--var-terms template variables 'terms'

ms_meta_biocrates

When specifying the –var-* argument, you can use JSON syntax. Failing to parse JSON will keep the string value.

```
cubi-tk isa-tpl ms_meta_biocrates [-h]
                                  [--var-investigation-title VAR_INVESTIGATION_TITLE]
                                  [--var-i-dir-name VAR_I_DIR_NAME]
                                  [--var-study-title VAR_STUDY_TITLE]
                                  [--var-study-id VAR_STUDY_ID]
                                  [--var-study-file-name VAR_STUDY_FILE_NAME]
                                  [--var-sample-names VAR_SAMPLE_NAMES]
                                  [--var-organism VAR_ORGANISM]
                                  [--var-organisms VAR_ORGANISMS]
                                  [--var-assay-measurement-type VAR_ASSAY_MEASUREMENT_
↪TYPE]
```

(continues on next page)

(continued from previous page)

```
                                    [--var-assay-technology-type VAR_ASSAY_TECHNOLOGY_
↪TYPE]
                                    [--var-assay-technology-types VAR_ASSAY_TECHNOLOGY_
↪TYPES]
                                    [--var-biocrates-kit VAR_BIOCRATES_KIT]
                                    [--var-assay-prefix VAR_ASSAY_PREFIX]
                                    [--var-assay-name VAR_ASSAY_NAME]
                                    [--var-assay-measurement-abbreviation-LC VAR_ASSAY_
↪MEASUREMENT_ABBREVIATION_LC]
                                    [--var-assay-measurement-abbreviation-FIA VAR_ASSAY_
↪MEASUREMENT_ABBREVIATION_FIA]
                                    [--var-biocrates-metidq-version VAR_BIOCRATES_
↪METIDQ_VERSION]
                                    [--var-metaquac-version VAR_METAQUAC_VERSION]
                                    [--var-instrument VAR_INSTRUMENT]
                                    [--var-instruments VAR_INSTRUMENTS]
                                    [--var-chromatography-instrument VAR_CHROMATOGRAPHY_
↪INSTRUMENT]
                                    output_dir
```

Positional Arguments

output_dir Path to output directory

Named Arguments

--var-investigation-title template variables 'investigation_title'

--var-i-dir-name template variables 'i_dir_name'

--var-study-title template variables 'study_title'

--var-study-id template variables 'study_id'

--var-study-file-name template variables 'study_file_name'

--var-sample-names template variables 'sample_names'

--var-organism template variables 'organism'

--var-organisms template variables 'organisms'

--var-assay-measurement-type template variables 'assay_measurement_type'

--var-assay-technology-type template variables 'assay_technology_type'

--var-assay-technology-types template variables 'assay_technology_types'

--var-biocrates-kit template variables 'biocrates_kit'

--var-assay-prefix template variables 'assay_prefix'

--var-assay-name template variables 'assay_name'

--var-assay-measurement-abbreviation-LC template variables 'assay_measurement_abbreviation_LC'

--var-assay-measurement-abbreviation-FIA template variables 'assay_measurement_abbreviation_FIA'

--var-biocrates-metidq-version template variables 'biocrates_metidq_version'

--var-metaquac-version template variables 'metaquac_version'

--var-instrument template variables 'instrument'

--var-instruments template variables 'instruments'

--var-chromatography-instrument template variables 'chromatography_instrument'

2.4.2 isa-tab

ISA-tab tools besides templating.

```
cubi-tk isa-tab [-h] {add-ped,resolve-hpo,annotate,validate} ...
```

Positional Arguments

isa_tab_cmd Possible choices: add-ped, resolve-hpo, annotate, validate

Sub-commands:

add-ped

Add records from PED file to ISA-tab

```
cubi-tk isa-tab add-ped [-h] [--sample-name-normalization {snappy,none}]
                        [--yes] [--dry-run] [--no-show-diff]
                        [--show-diff-side-by-side] [--batch-no BATCH_NO]
                        [--library-type {WES,WGS,Panel_seq}]
                        [--library-layout {SINGLE,PAIRED}]
                        [--library-kit LIBRARY_KIT]
                        [--library-kit-catalogue-id LIBRARY_KIT_CATALOGUE_ID]
                        [--platform PLATFORM]
                        [--instrument-model INSTRUMENT_MODEL]
                        investigation.tsv pedigree.ped
```

Positional Arguments

investigation.tsv Path to ISA-tab investigation file.

pedigree.ped Path to PLINK PED file with records to add.

Named Arguments

--sample-name-normalization Possible choices: snappy, none

Normalize sample names, default: snappy, choices: snappy, none

Default: "snappy"

--yes Assume all answers are yes.

Default: False

--dry-run, -n Perform a dry run, i.e., don't change anything only display change, implies '–show-diff'.

Default: False

--no-show-diff, -D Don't show change when creating/updating sample sheets.

Default: True

--show-diff-side-by-side Show diff side by side instead of unified.

Default: False

--batch-no Value to set as the batch number.

Default: "."

--library-type Possible choices: WES, WGS, Panel_seq

The library type.

Default: "WES"

--library-layout Possible choices: SINGLE, PAIRED

The library layout.

Default: "PAIRED"

--library-kit The library kit used.

Default: ""

--library-kit-catalogue-id The library kit catalogue ID.

Default: ""

--platform The string to use for the platform

Default: "ILLUMINA"

--instrument-model The string to use for the instrument model

Default: ""

resolve-hpo

Resolve HPO term lists to ISA-tab fragments

```
cubi-tk isa-tab resolve-hpo [-h] [--hpo-obo-url HPO_OBO_URL] [term_file]
```

Positional Arguments

term_file Path to ISA-tab investigation file.

Default: <_io.TextIOWrapper name='<stdin>' mode='r' encoding='UTF-8'>

Named Arguments

--hpo-obo-url Default URL to OBO file.

Default: "http://purl.obolibrary.org/obo/hp.obo"

annotate

Add annotation from CSV file to ISA-tab

```
cubi-tk isa-tab annotate [-h] [--yes] [--dry-run] [--no-show-diff]
                         [--show-diff-side-by-side] [--force-update]
                         [--target-study s_study.tsv]
                         [--target-assay a_assay.tsv]
                         investigation.tsv annotation.tsv
```

Positional Arguments

investigation.tsv Path to ISA-tab investigation file.

annotation.tsv Path to annotation (TSV) file with information to add.

Named Arguments

--yes Assume all answers are yes.

Default: False

--dry-run, -n Perform a dry run, i.e., don't change anything only display change, implies '–show-diff'.

Default: False

--no-show-diff, -D Don't show change when creating/updating sample sheets.

Default: True

--show-diff-side-by-side Show diff side by side instead of unified.

Default: False

--force-update Overwrite non-empty ISA-tab entries.

Default: False

--target-study, -s File name study to annotate. If not provided, first study in investigation is used.

--target-assay, -a File name of assay to annotate. If not provided, first assay in investigation is used.

validate

Validate ISA-tab

```
cubi-tk isa-tab validate [-h] [--show-duplicate-warnings] investigation.tsv
```

Positional Arguments

investigation.tsv Path to ISA-tab investigation file.

Named Arguments

--show-duplicate-warnings Show duplicated warnings, i.e. with same message and same category (False by default)

Default: False

2.4.3 snappy

Tools for supporting the SNAPPY pipeline.

```
cubi-tk snappy [-h]
               {check,itransfer-raw-data,itransfer-ngs-mapping,itransfer-variant-
↪calling,pull-sheets,pull-raw-data,varfish-upload,kickoff}
               ...
```

Positional Arguments

snappy_cmd Possible choices: check, itransfer-raw-data, itransfer-ngs-mapping, itransfer-variant-calling, pull-sheets, pull-raw-data, varfish-upload, kickoff

Sub-commands:

check

Check consistency within sample sheet and between sheet and files

```
cubi-tk snappy check [-h] [--tsv-shortcut {germline,cancer}]
                     [--base-path BASE_PATH]
                     biomedsheet_tsv [biomedsheet_tsv ...]
```

Positional Arguments

biomedsheet_tsv Path to biomedsheets TSV file to load.

Named Arguments

--tsv-shortcut Possible choices: germline, cancer

The shortcut TSV schema to use.

Default: "germline"

--base-path Base path of project (contains 'ngs_mapping/' etc.), spiders up from biomedsheet_tsv and falls back to current working directory by default.

itransfer-raw-data

Transfer FASTQs into iRODS landing zone

```
cubi-tk snappy itransfer-raw-data [-h] [--sodar-url SODAR_URL]
                                  [--sodar-api-token SODAR_API_TOKEN]
                                  [--num-parallel-transfers NUM_PARALLEL_TRANSFERS]
                                  [--tsv-shortcut {germline,cancer}]
                                  [--start-batch START_BATCH]
                                  [--base-path BASE_PATH]
                                  [--remote-dir-date REMOTE_DIR_DATE]
                                  [--remote-dir-pattern REMOTE_DIR_PATTERN]
                                  [--yes] [--validate-and-move]
                                 biomedsheet_tsv destination
```

Positional Arguments

biomedsheet_tsv Path to biomedsheets TSV file to load.

destination UUID or iRods path of landing zone to move to.

Named Arguments

--num-parallel-transfers Number of parallel transfers, defaults to 8

Default: 8

--tsv-shortcut Possible choices: germline, cancer

The shortcut TSV schema to use.

Default: "germline"

--start-batch Batch to start the transfer at, defaults to 0.

Default: 0

--base-path Base path of project (contains 'ngs_mapping/' etc.), defaults to current path.

Default: "/home/docs/checkouts/readthedocs.org/user_builds/cubi-tk/checkouts/stable/docs_manual"

--remote-dir-date Date to use in remote directory, defaults to YYYY-MM-DD of today.

Default: "2021-05-05"

--remote-dir-pattern Pattern to use for constructing remote pattern

Default: "{library_name}/raw_data/{date}"

--yes Assume all answers are yes, e.g., will create or use existing available landing zones without asking.

Default: False

--validate-and-move After files are transferred to SODAR, it will proceed with validation and move.

Default: False

SODAR-related

--sodar-url URL to SODAR, defaults to SODAR_URL environment variable or fallback to https://sodar.bihealth.org/

Default: "https://sodar.bihealth.org/"

--sodar-api-token Authentication token when talking to SODAR. Defaults to SODAR_API_TOKEN environment variable.

itransfer-ngs-mapping

Transfer ngs_mapping results into iRODS landing zone

```
cubi-tk snappy itransfer-ngs-mapping [-h] [--sodar-url SODAR_URL]
                                     [--sodar-api-token SODAR_API_TOKEN]
                                     [--num-parallel-transfers NUM_PARALLEL_TRANSFERS]
                                     [--tsv-shortcut {germline,cancer}]
                                     [--start-batch START_BATCH]
                                     [--base-path BASE_PATH]
                                     [--remote-dir-date REMOTE_DIR_DATE]
                                     [--remote-dir-pattern REMOTE_DIR_PATTERN]
                                     [--yes] [--validate-and-move]
                                     [--mapper MAPPER]
                                    biomedsheet_tsv destination
```

Positional Arguments

biomedsheet_tsv Path to biomedsheets TSV file to load.

destination UUID or iRods path of landing zone to move to.

Named Arguments

--num-parallel-transfers Number of parallel transfers, defaults to 8

Default: 8

--tsv-shortcut Possible choices: germline, cancer

The shortcut TSV schema to use.

Default: "germline"

--start-batch Batch to start the transfer at, defaults to 0.

Default: 0

--base-path Base path of project (contains 'ngs_mapping/' etc.), defaults to current path.

Default: "/home/docs/checkouts/readthedocs.org/user_builds/cubi-tk/checkouts/stable/docs_manual"

--remote-dir-date Date to use in remote directory, defaults to YYYY-MM-DD of today.

Default: "2021-05-05"

--remote-dir-pattern Pattern to use for constructing remote pattern

Default: "{library_name}/ngs_mapping/{date}"

--yes Assume all answers are yes, e.g., will create or use existing available landing zones without asking.

Default: False

--validate-and-move After files are transferred to SODAR, it will proceed with validation and move.

Default: False

--mapper Name of the mapper to transfer for, defaults to bwa.

Default: "bwa"

SODAR-related

--sodar-url URL to SODAR, defaults to SODAR_URL environment variable or fallback to https://sodar.bihealth.org/

Default: "https://sodar.bihealth.org/"

--sodar-api-token Authentication token when talking to SODAR. Defaults to SODAR_API_TOKEN environment variable.

itransfer-variant-calling

Transfer variant_calling results into iRODS landing zone

```
cubi-tk snappy itransfer-variant-calling [-h] [--sodar-url SODAR_URL]
                                         [--sodar-api-token SODAR_API_TOKEN]
                                         [--num-parallel-transfers NUM_PARALLEL_
↪TRANSFERS]
                                         [--tsv-shortcut {germline,cancer}]
                                         [--start-batch START_BATCH]
                                         [--base-path BASE_PATH]
                                         [--remote-dir-date REMOTE_DIR_DATE]
                                         [--remote-dir-pattern REMOTE_DIR_PATTERN]
                                         [--yes] [--validate-and-move]
                                         [--mapper MAPPER] [--caller CALLER]
                                        biomedsheet_tsv destination
```

Positional Arguments

biomedsheet_tsv Path to biomedsheets TSV file to load.

destination UUID or iRods path of landing zone to move to.

Named Arguments

--num-parallel-transfers Number of parallel transfers, defaults to 8

Default: 8

--tsv-shortcut Possible choices: germline, cancer

The shortcut TSV schema to use.

Default: "germline"

--start-batch Batch to start the transfer at, defaults to 0.

Default: 0

--base-path Base path of project (contains 'ngs_mapping/' etc.), defaults to current path.

Default: "/home/docs/checkouts/readthedocs.org/user_builds/cubi-tk/checkouts/stable/docs_manual"

--remote-dir-date Date to use in remote directory, defaults to YYYY-MM-DD of today.

Default: "2021-05-05"

--remote-dir-pattern Pattern to use for constructing remote pattern

Default: "{library_name}/variant_calling/{date}"

--yes Assume all answers are yes, e.g., will create or use existing available landing zones without asking.

Default: False

--validate-and-move After files are transferred to SODAR, it will proceed with validation and move.

Default: False

--mapper Name of the mapper to transfer for, defaults to bwa.

Default: "bwa"

--caller Name of the variant caller to transfer for, defaults to gatk_hc

Default: "gatk_hc"

SODAR-related

--sodar-url URL to SODAR, defaults to SODAR_URL environment variable or fallback to https://sodar.bihealth.org/

Default: "https://sodar.bihealth.org/"

--sodar-api-token Authentication token when talking to SODAR. Defaults to SODAR_API_TOKEN environment variable.

pull-sheets

Pull SODAR sample sheets into biomedsheet

```
cubi-tk snappy pull-sheets [-h] [--base-path BASE_PATH] [--yes] [--dry-run]
                           [--no-show-diff] [--show-diff-side-by-side]
                           [--library-types LIBRARY_TYPES]
```

Named Arguments

--base-path Base path of project (contains '.snappy_pipeline/' etc.), spiders up from current work directory and falls back to current working directory by default.

Default: "/home/docs/checkouts/readthedocs.org/user_builds/cubi-tk/checkouts/stable/docs_manual"

--yes Assume all answers are yes.

Default: False

--dry-run, -n Perform a dry run, i.e., don't change anything only display change, implies '–show-diff'.

Default: False

--no-show-diff, -D Don't show change when creating/updating sample sheets.

Default: True

--show-diff-side-by-side Show diff side by side instead of unified.

Default: False

--library-types Library type(s) to use, comma-separated, default is to use all.

pull-raw-data

Pull raw data from SODAR to SNAPPY dataset raw data directory

```
cubi-tk snappy pull-raw-data [-h] [--base-path BASE_PATH]
                             [--sodar-url SODAR_URL]
                             [--sodar-api-token SODAR_API_TOKEN] [--overwrite]
                             [--min-batch MIN_BATCH] [--samples SAMPLES]
                             [--yes] [--dry-run]
                             [--irsync-threads IRSYNC_THREADS] [--assay ASSAY]
                             project_uuid
```

Positional Arguments

project_uuid UUID of project to download data for.

Named Arguments

--base-path Base path of project (contains '.snappy_pipeline/' etc.), spiders up from current work directory and falls back to current working directory by default.

Default: "/home/docs/checkouts/readthedocs.org/user_builds/cubi-tk/checkouts/stable/docs_manual"

--overwrite Allow overwriting of files

Default: False

--min-batch Minimal batch number to pull

Default: 0

--samples Optional list of samples to pull

--yes Assume all answers are yes.

Default: False

--dry-run, -n Perform a dry run, i.e., don't change anything only display change, implies '–show-diff'.

Default: False

--irsync-threads Parameter -N to pass to irsync

--assay UUID of assay to create landing zone for.

SODAR-related

--sodar-url URL to SODAR, defaults to SODAR_URL environment variable or fallback to https://sodar.bihealth.org/

Default: "https://sodar.bihealth.org/"

--sodar-api-token Authentication token when talking to SODAR. Defaults to SODAR_API_TOKEN environment variable.

varfish-upload

Upload variant analysis results into VarFish

```
cubi-tk snappy varfish-upload [-h] [--varfish-config VARFISH_CONFIG]
                              [--varfish-server-url VARFISH_SERVER_URL]
                              [--varfish-api-token VARFISH_API_TOKEN]
                              [--base-path BASE_PATH] [--steps STEPS]
                              [--min-batch MIN_BATCH] [--yes]
                              [--samples SAMPLES]
                              project [project ...]
```

Positional Arguments

project The UUID(s) of the SODAR project to submit.

Named Arguments

--base-path Base path of project (contains '.snappy_pipeline/' etc.), spiders up from current work directory and falls back to current working directory by default.

Default: "/home/docs/checkouts/readthedocs.org/user_builds/cubi-tk/checkouts/stable/docs_manual"

--steps Pipeline steps to consider for the export. Defaults to include all of the following; specify this with +name/-name to add/remove and either give multiple arguments or use a comma-separated list. {ngs_mapping, targeted_seq_cnv_export, variant_export, wgs_cnv_export, wgs_sv_export}

Default: []

--min-batch	Smallest batch to transfer, keep empty to transfer all.
--yes, -y	Assume yes to all answers Default: False
--samples	The samples to limit the submission for, if any Default: “”

VarFish Configuration

--varfish-config	Path to configuration file.
--varfish-server-url	SODAR server URL key to use, defaults to env VARFISH_SERVER_URL.
--varfish-api-token	SODAR API token to use, defaults to env VARFISH_API_TOKEN.

kickoff

Kick-off SNAPPY pipeline steps.

```
cubi-tk snappy kickoff [-h] [--dry-run] [--timeout TIMEOUT] [path]
```

Positional Arguments

path	Path into SNAPPY directory (below a directory containing .snappy_pipeline).

Named Arguments

--dry-run, -n	Perform dry-run, do not do anything. Default: False
--timeout	Number of seconds to wait for commands. Default: 10

2.4.4 sodar

SODAR command line interface.

```
cubi-tk sodar [-h]
              {add-ped,download-sheet,upload-sheet,pull-raw-data,landing-zone-create,
↪landing-zone-list,landing-zone-move,ingest-fastq}
              ...
```

Positional Arguments

sodar_cmd	Possible choices: add-ped, download-sheet, upload-sheet, pull-raw-data, landing-zone-create, landing-zone-list, landing-zone-move, ingest-fastq

Sub-commands:

add-ped

Augment sample sheet from PED file

```
cubi-tk sodar add-ped [-h] [--sodar-url SODAR_URL]
                      [--sodar-api-token SODAR_API_TOKEN] [--dry-run]
                      [--show-diff] [--show-diff-side-by-side]
                      [--sample-name-normalization {snappy,none}] [--yes]
                      [--batch-no BATCH_NO]
                      [--library-type {WES,WGS,Panel_seq}]
                      [--library-layout {SINGLE,PAIRED}]
                      [--library-kit LIBRARY_KIT]
                      [--library-kit-catalogue-id LIBRARY_KIT_CATALOGUE_ID]
                      [--platform PLATFORM]
                      [--instrument-model INSTRUMENT_MODEL]
                      project_uuid pedigree.ped
```

Positional Arguments

project_uuid UUID of project to download the ISA-tab for.

pedigree.ped Path to PLINK PED file with records to add.

Named Arguments

--dry-run, -n Perform a dry run, i.e., don't change anything only display change, implies '–show-diff'.

Default: False

--show-diff, -D Show change when creating/updating sample sheets.

Default: False

--show-diff-side-by-side Show diff side by side instead of unified.

Default: False

--sample-name-normalization Possible choices: snappy, none

Normalize sample names, default: snappy, choices: snappy, none

Default: "snappy"

--yes Assume all answers are yes.

Default: False

--batch-no Value to set as the batch number.

Default: "."

--library-type Possible choices: WES, WGS, Panel_seq

The library type.

Default: "WES"

--library-layout Possible choices: SINGLE, PAIRED

The library layout.

Default: "PAIRED"

--library-kit The library kit used.

Default: ""

--library-kit-catalogue-id The library kit catalogue ID.

Default: ""

--platform The string to use for the platform

Default: "ILLUMINA"

--instrument-model The string to use for the instrument model

Default: ""

SODAR-related

--sodar-url URL to SODAR, defaults to SODAR_URL environment variable or fallback to https://sodar.bihealth.org/

Default: "https://sodar.bihealth.org/"

--sodar-api-token Authentication token when talking to SODAR. Defaults to SODAR_API_TOKEN environment variable.

download-sheet

Download ISA-tab

```
cubi-tk sodar download-sheet [-h] [--sodar-url SODAR_URL]
                             [--sodar-api-token SODAR_API_TOKEN]
                             [--no-makedirs] [--overwrite] [--yes] [--dry-run]
                             [--show-diff] [--show-diff-side-by-side]
                             project_uuid output_dir
```

Positional Arguments

project_uuid UUID of project to download the ISA-tab for.

output_dir Path to output directory to write the sheet to.

Named Arguments

--no-makedirs Create output directories

Default: True

--overwrite Allow overwriting of files

Default: False

--yes Assume all answers are yes.

Default: False

--dry-run, -n Perform a dry run, i.e., don't change anything only display change, implies '–show-diff'.

Default: False

--show-diff, -D Show change when creating/updating sample sheets.

Default: False

--show-diff-side-by-side Show diff side by side instead of unified.

Default: False

SODAR-related

--sodar-url URL to SODAR, defaults to SODAR_URL environment variable or fallback to https://sodar.bihealth.org/

Default: "https://sodar.bihealth.org/"

--sodar-api-token Authentication token when talking to SODAR. Defaults to SODAR_API_TOKEN environment variable.

upload-sheet

Upload and replace ISA-tab

```
cubi-tk sodar upload-sheet [-h] [--sodar-url SODAR_URL]
                           [--sodar-api-token SODAR_API_TOKEN]
                           project_uuid input_investigation_file
```

Positional Arguments

project_uuid UUID of project to upload the ISA-tab for.

input_investigation_file Path to input investigation file.

SODAR-related

--sodar-url URL to SODAR, defaults to SODAR_URL environment variable or fallback to https://sodar.bihealth.org/

Default: "https://sodar.bihealth.org/"

--sodar-api-token Authentication token when talking to SODAR. Defaults to SODAR_API_TOKEN environment variable.

pull-raw-data

Download raw data from iRODS

```
cubi-tk sodar pull-raw-data [-h] [--sodar-url SODAR_URL]
                            [--sodar-api-token SODAR_API_TOKEN] [--overwrite]
                            [--min-batch MIN_BATCH] [--yes] [--dry-run]
                            [--irsync-threads IRSYNC_THREADS] [--assay ASSAY]
                           project_uuid output_dir
```

Positional Arguments

project_uuid	UUID of project to download data for.
output_dir	Path to output directory to write the raw data to.

Named Arguments

--overwrite	Allow overwriting of files Default: False
--min-batch	Minimal batch number to pull Default: 0
--yes	Assume all answers are yes. Default: False
--dry-run, -n	Perform a dry run, i.e., don't change anything only display change, implies '–show-diff'. Default: False
--irsync-threads	Parameter -N to pass to irsync
--assay	UUID of assay to download data for.

SODAR-related

--sodar-url	URL to SODAR, defaults to SODAR_URL environment variable or fallback to https://sodar.bihealth.org/ Default: "https://sodar.bihealth.org/"
--sodar-api-token	Authentication token when talking to SODAR. Defaults to SODAR_API_TOKEN environment variable.

landing-zone-create

Creating landing zone

```
cubi-tk sodar landing-zone-create [-h] [--sodar-url SODAR_URL]
                                  [--sodar-api-token SODAR_API_TOKEN]
                                  [--unless-exists] [--dry-run]
                                  [--assay ASSAY] [--format FORMAT_STRING]
                                  project_uuid
```

Positional Arguments

project_uuid UUID of project to create the landing zone in.

Named Arguments

--unless-exists If there already is a landing zone in the current project then use this one

Default: False

--dry-run, -n Perform a dry run, i.e., don't change anything only display change, implies '–show-diff'.

Default: False

--assay UUID of assay to create landing zone for.

--format Format string for printing, e.g. %(uuid)s

SODAR-related

--sodar-url URL to SODAR, defaults to SODAR_URL environment variable or fallback to https://sodar.bihealth.org/

Default: "https://sodar.bihealth.org/"

--sodar-api-token Authentication token when talking to SODAR. Defaults to SODAR_API_TOKEN environment variable.

landing-zone-list

List landing zones

```
cubi-tk sodar landing-zone-list [-h] [--sodar-url SODAR_URL]
                                [--sodar-api-token SODAR_API_TOKEN]
                                [--unless-exists] [--dry-run]
                                [--format FORMAT_STRING]
                                project_uuid
```

Positional Arguments

project_uuid UUID of project to create the landing zone in.

Named Arguments

--unless-exists If there already is a landing zone in the current project then use this one

Default: False

--dry-run, -n Perform a dry run, i.e., don't change anything only display change, implies '–show-diff'.

Default: False

--format Format string for printing, e.g. %(uuid)s

SODAR-related

--sodar-url URL to SODAR, defaults to SODAR_URL environment variable or fallback to https://sodar.bihealth.org/

Default: "https://sodar.bihealth.org/"

--sodar-api-token Authentication token when talking to SODAR. Defaults to SODAR_API_TOKEN environment variable.

landing-zone-move

Submit landing zone for moving

```
cubi-tk sodar landing-zone-move [-h] [--sodar-url SODAR_URL]
                                [--sodar-api-token SODAR_API_TOKEN]
                                [--dry-run] [--format FORMAT_STRING]
                                landing_zone_uuid
```

Positional Arguments

landing_zone_uuid UUID of landing zone to move.

Named Arguments

--dry-run, -n Perform a dry run, i.e., don't change anything only display change, implies '–show-diff'.

Default: False

--format Format string for printing, e.g. %(uuid)s

SODAR-related

--sodar-url URL to SODAR, defaults to SODAR_URL environment variable or fallback to https://sodar.bihealth.org/

Default: "https://sodar.bihealth.org/"

--sodar-api-token Authentication token when talking to SODAR. Defaults to SODAR_API_TOKEN environment variable.

ingest-fastq

Upload external files to SODAR (defaults for fastq)

```
cubi-tk sodar ingest-fastq [-h] [--sodar-url SODAR_URL]
                           [--sodar-api-token SODAR_API_TOKEN]
                           [--num-parallel-transfers NUM_PARALLEL_TRANSFERS]
                           [--yes] [--base-path BASE_PATH]
                           [--remote-dir-date REMOTE_DIR_DATE]
                           [--src-regex SRC_REGEX]
                           [--remote-dir-pattern REMOTE_DIR_PATTERN]
                           [--add-suffix ADD_SUFFIX] [-m MATCH REPL]
                           [--tmp TMP]
                           sources [sources ...] destination
```

Positional Arguments

sources paths to fastq folders

destination UUID or iRods path of landing zone to move to.

Named Arguments

--num-parallel-transfers Number of parallel transfers, defaults to 8

Default: 8

--yes Assume the answer to all prompts is 'yes'

Default: False

--base-path Base path of project (contains 'ngs_mapping/' etc.), defaults to current path.

Default: "/home/docs/checkouts/readthedocs.org/user_builds/cubi-tk/checkouts/stable/docs_manual"

--remote-dir-date Date to use in remote directory, defaults to YYYY-MM-DD of today.

Default: "2021-05-05"

--src-regex Regular expression to use for matching input fastq files, default: (.*/)?(?P<sample>.+?)(?:_(?P<lane>L[0-9]+?))?(?:_(?P<mate>R[0-9]+?))?(?:_(?P<batch>[0-9]+?))?.f(?:ast)?q.gz

Default: "(.*/)?(?P<sample>.+?)(?:_(?P<lane>L[0-9]+?))?(?:_(?P<mate>R[0-9]+?))?(?:_(?P<batch>[0-9]+?))?.f(?:ast)?q.gz"

--remote-dir-pattern Pattern to use for constructing remote pattern, default: {sample}/{date}/{filename}

Default: "{sample}/{date}/{filename}"

--add-suffix Suffix to add to all file names (e.g. '-N1-DNA1-WES1').

Default: ""

-m, --remote-dir-mapping Substitutions applied to the filled remote dir paths. Can for example be used to modify sample names. Use pythons regex syntax of 're.sub' package. This argument can be used multiple times (i.e. '-m <regex1> <repl1> -m <regex2> <repl2>' ...).

Default: []

--tmp Folder to save files from WebDAV temporarily, if set as source.

Default: "temp/"

SODAR-related

--sodar-url URL to SODAR, defaults to SODAR_URL environment variable or fallback to https://sodar.bihealth.org/

Default: "https://sodar.bihealth.org/"

--sodar-api-token Authentication token when talking to SODAR. Defaults to SODAR_API_TOKEN environment variable.

2.4.5 irods

iRods command line interface.

```
cubi-tk irods [-h] {check} ...
```

Positional Arguments

irods_cmd Possible choices: check

Sub-commands:

check

Check target iRods collection (all md5 files? metadata md5 consistent? enough replicas?).

```
cubi-tk irods check [-h] [--num-replicas NUM_REPLICAS]
                    [--num-parallel-tests NUM_PARALLEL_TESTS]
                    irods_path
```

Positional Arguments

irods_path Path to an iRods collection.

Named Arguments

--num-replicas Minimum number of replicas, defaults to 2

Default: 2

--num-parallel-tests Number of parallel tests, defaults to 8

Default: 8

2.4.6 org-raw

org_raw command line interface.

```
cubi-tk org-raw [-h] {check,organize} ...
```

Positional Arguments

org_raw_cmd Possible choices: check, organize

Sub-commands:

check

Check consistency of raw data

```
cubi-tk org-raw check [-h] [--num-threads NUM_THREADS] [--no-gz-check]
                      [--no-md5-check] [--no-compute-md5]
                      [--missing-md5-error] [--create-md5-fail-no-error]
                      FILE.fastq.gz [FILE.fastq.gz ...]
```

Positional Arguments

FILE.fastq.gz Path(s) to .fastq.gz files to perform the check for

Named Arguments

--num-threads Number of parallel threads

Default: 0

--no-gz-check Deactivate check for gzip consistency (default is to perform check).

Default: True

--no-md5-check Deactivate comparison of MD5 sum if .md5 file exists (default is to perform check).

Default: True

--no-compute-md5 Deactivate computation of MD5 sum if missing (default is to compute MD5 sum).

Default: True

--missing-md5-error Make missing .md5 files constitute an error. Default is to issue an log message only.

Default: False

--create-md5-fail-no-error Make failure to create .md5 file not an error. Default is to make it an error.

Default: True

organize

Check consistency of raw data

```
cubi-tk org-raw organize [-h] [--dry-run] [--yes] [--move] [--no-check]
                         [--src-regex SRC_REGEX] [--dest-pattern DEST_PATTERN]
                         [--num-threads NUM_THREADS] [--no-gz-check]
                         [--no-md5-check] [--no-compute-md5]
                         [--missing-md5-error] [--create-md5-fail-no-error]
                         out_path path.fastq.gz [path.fastq.gz ...]
```

Positional Arguments

out_path Path to output directory.

path.fastq.gz Path to input files.

Named Arguments

--dry-run Dry-run, do not actually do anything

Default: False

--yes Assume the answer to all prompts is 'yes'

Default: False

--move Move file(s) instead of copying, default is to copy.

Default: False

--no-check Do not run 'raw-org check' on output (default is to run).

Default: True

--src-regex Regular expression for parsing file paths. Default: (.*/)?(?P<sample>.+)(?:-.+?)?.f(?:ast)?q.gz

Default: "(.*/)?(?P<sample>.+)(?:-.+?)?.f(?:ast)?q.gz"

--dest-pattern Format expression for destination path generation. Default: {sample_name}/{file_name}

Default: "{sample_name}/{file_name}"

--num-threads Number of parallel threads

Default: 0

--no-gz-check Deactivate check for gzip consistency (default is to perform check).

Default: True

--no-md5-check Deactivate comparison of MD5 sum if .md5 file exists (default is to perform check).

Default: True

--no-compute-md5 Deactivate computation of MD5 sum if missing (default is to compute MD5 sum).

Default: True

--missing-md5-error Make missing .md5 files constitute an error. Default is to issue an log message only.

Default: False

--create-md5-fail-no-error Make failure to create .md5 file not an error. Default is to make it an error.

Default: True

2.4.7 sea-snap

Tools for supporting the RNA-SeASnaP pipeline.

```
cubi-tk sea-snap [-h]
                 {itransfer-raw-data,itransfer-results,working-dir,write-sample-info,
↪check-irods}
                 ...
```

Positional Arguments

sea_snap_cmd Possible choices: itransfer-raw-data, itransfer-results, working-dir, write-sample-info, check-irods

Sub-commands:

itransfer-raw-data

Transfer FASTQs into iRODS landing zone

```
cubi-tk sea-snap itransfer-raw-data [-h] [--sodar-url SODAR_URL]
                                    [--sodar-api-token SODAR_API_TOKEN]
                                    [--num-parallel-transfers NUM_PARALLEL_TRANSFERS]
                                    [--tsv-shortcut {germline,cancer}]
                                    [--start-batch START_BATCH]
                                    [--base-path BASE_PATH]
                                    [--remote-dir-date REMOTE_DIR_DATE]
                                    [--remote-dir-pattern REMOTE_DIR_PATTERN]
                                    [--yes] [--validate-and-move]
                                   biomedsheet_tsv destination
```

Positional Arguments

biomedsheet_tsv Path to biomedsheets TSV file to load.

destination UUID or iRods path of landing zone to move to.

Named Arguments

--num-parallel-transfers Number of parallel transfers, defaults to 8

Default: 8

--tsv-shortcut Possible choices: germline, cancer

The shortcut TSV schema to use.

Default: "germline"

--start-batch Batch to start the transfer at, defaults to 0.

Default: 0

--base-path Base path of project (contains 'ngs_mapping/' etc.), defaults to current path.

Default: "/home/docs/checkouts/readthedocs.org/user_builds/cubi-tk/checkouts/stable/docs_manual"

--remote-dir-date Date to use in remote directory, defaults to YYYY-MM-DD of today.

Default: "2021-05-05"

--remote-dir-pattern Pattern to use for constructing remote pattern

Default: "{library_name}/raw_data/{date}"

--yes Assume all answers are yes, e.g., will create or use existing available landing zones without asking.

Default: False

--validate-and-move After files are transferred to SODAR, it will proceed with validation and move.

Default: False

SODAR-related

--sodar-url URL to SODAR, defaults to SODAR_URL environment variable or fallback to https://sodar.bihealth.org/

Default: "https://sodar.bihealth.org/"

--sodar-api-token Authentication token when talking to SODAR. Defaults to SODAR_API_TOKEN environment variable.

itransfer-results

Transfer mapping results into iRODS landing zone

```
cubi-tk sea-snap itransfer-results [-h] [--sodar-url SODAR_URL]
                                   [--sodar-api-token SODAR_API_TOKEN]
                                   [--num-parallel-transfers NUM_PARALLEL_TRANSFERS]
                                   transfer_blueprint destination
```

Positional Arguments

transfer_blueprint Path to blueprint file to load. This file contains commands to sync files with iRODS. Blocks of commands separated by an empty line will be executed together in one thread.

destination UUID or iRods path of landing zone to move to.

Named Arguments

--num-parallel-transfers Number of parallel transfers, defaults to 8

Default: 8

SODAR-related

--sodar-url URL to SODAR, defaults to SODAR_URL environment variable or fallback to https://sodar.bihealth.org/

Default: "https://sodar.bihealth.org/"

--sodar-api-token Authentication token when talking to SODAR. Defaults to SODAR_API_TOKEN environment variable.

working-dir

Create working directory

```
cubi-tk sea-snap working-dir [-h] [--dry-run] [--dirname DIRNAME]
                             [--configs {mapping,DE} [{mapping,DE} ...]]
                             [sea_snap_path]
```

Positional Arguments

sea_snap_path Path into RNA-SeA-SnaP directory (below a directory containing 'mapping_pipeline.snake').

Default: "/home/docs/checkouts/readthedocs.org/user_builds/cubi-tk/checkouts/stable/docs_manual"

Named Arguments

--dry-run, -n Perform dry-run, do not do anything.

Default: False

--dirname, -d Name of the working directory to create (default: 'results_YEAR_MONTH_DAY/').

Default: "results_%Y_%m_%d/"

--configs, -c Possible choices: mapping, DE

Configs to be imported (default: all).

Default: ['mapping', 'DE']

write-sample-info

Generate sample info file

```
cubi-tk sea-snap write-sample-info [-h] [--allow-overwrite] [--dry-run]
                                   [--show-diff] [--show-diff-side-by-side]
                                   [--from-file FROM_FILE]
                                   [--isa-assay ISA_ASSAY]
                                   [--project_uuid PROJECT_UUID]
                                   [--output_folder OUTPUT_FOLDER]
                                   [--overwrite-isa] [--sodar-url SODAR_URL]
                                   [--sodar-auth-token SODAR_AUTH_TOKEN]
                                  in_path_pattern [output_file]
```

Positional Arguments

in_path_pattern Path pattern to use for extracting input file information. See https://cubi-gitlab.bihealth.org/CUBI/Pipelines/sea-snap/blob/master/documentation/prepare_input.md#fastq-files-folder-structure.

output_file Filename ending with '.yaml' or '.tsv'. default: sample_info.yaml.

Default: sample_info.yaml

Named Arguments

--allow-overwrite Allow to overwrite output file, default is not to allow overwriting output file.

Default: False

--dry-run Perform a dry run, i.e., don't change anything only display change, implies '–show-diff'.

Default: False

--show-diff Show change when creating/updating sample sheets.

Default: False

--show-diff-side-by-side Show diff side by side instead of unified.

Default: False

--from-file Path to yaml file to convert to tsv or tsv to yaml. Not used, if not specified.

--isa-assay Path to ISA assay file. Not used, if not specified.

pull ISA files

--project_uuid If set pull ISA files from SODAR. UUID of project to pull from.

Default: False

--output_folder Output folder path to store ISA files.

Default: "ISA_files/"

--overwrite-isa Allow to overwrite output file, default is not to allow overwriting output file.

Default: False

SODAR-related

--sodar-url URL to SODAR, defaults to SODAR_URL environment variable or fallback to https://sodar.bihealth.org/

Default: "https://sodar.bihealth.org/"

--sodar-auth-token Authentication token when talking to SODAR. Defaults to SODAR_AUTH_TOKEN environment variable.

check-irods

Check consistency of sample info, blueprint and files on SODAR

```
cubi-tk sea-snap check-irods [-h] [--num-replicas NUM_REPLICAS]
                             [--num-parallel-tests NUM_PARALLEL_TESTS] [--yes]
                             [--transfer-blueprint TRANSFER_BLUEPRINT]
                             results_folder irods_path
```

Positional Arguments

results_folder Path to a Sea-snap results folder.

irods_path Path to an iRods collection.

Named Arguments

--num-replicas Minimum number of replicas, defaults to 2

Default: 2

--num-parallel-tests Number of parallel tests, defaults to 8

Default: 8

--yes Assume the answer to all prompts is 'yes'

Default: False

--transfer-blueprint Filename of blueprint file for export to SODAR (created e.g. with './sea-snap sc l export'). Assumed to be in the results folder. Default: 'SODAR_export_blueprint.txt'

Default: "SODAR_export_blueprint.txt"

CHAPTER 3

Manual for `isa-tpl`

`cubi-tk isa-tpl`: create ISA-tab directories using Cookiecutter.

You can use this command to quickly bootstrap an ISA-tab investigation. The functionality is built on Cookiecutter.

To create a directory with ISA-tab files, run:

```
$ cubi-tk isa-tpl <template name> <output directory>
```

This will prompt a number of questions interactively on the command line to collect information about the files that are going to be created. The requested information will depend on the chosen ISA-tab template. It is also possible to pass this information non-interactively together with other command line arguments (see `cubi-tk isa-tpl <template name> --help`).

The completed information will then be used to create a directory with ISA-tab files. It will be necessary to edit and extend the automatically generated files, e.g. to add additional rows to the assays.

3.1 Available Templates

The Cookiecutter directories are located in this module's directory. Currently available templates are:

- `isatab-generic`
- `isatab-germline`
- `isatab-microarray`
- `isatab-ms_meta_biocrates`
- `isatab-single_cell_rnaseq`
- `isatab-tumor_normal_dna`
- `isatab-tumor_normal_triplets`

3.2 Adding Templates

Adding templates consists of the following steps:

1. Add a new template directory below `cubi_tk/isa_tpl`.
2. Register it appending a `IsaTabTemplate` object to `_TEMPLATES` in `cubi_tk.isa_tpl`.
3. Add it to the list above in the docstring.

The easiest way to start out is to copy an existing cookiecutter template and registration.

3.3 More Information

Also see `cubi-tk isa-tpl` CLI documentation and `cubi-tk isa-tab --help` for more information.

CHAPTER 4

Manual for `isa-tab`

`cubi-tk isa-tab`: ISA-tab tooling.

4.1 Sub Commands

`validate` Validate ISA-tab files for correctness and perform sanity checks.

`resolve-hpo` Resolve lists of HPO terms to TSV suitable for copy-and-paste into ISA-tab.

`add-ped` Given a germline DNA sequencing ISA-tab file and a PED file, add new lines to the ISA-tab file and update existing ones, e.g., for newly added parents.

`annotate` Add annotation to an ISA-tab file, given a tsv file.

4.2 Annotate

`cubi-tk isa-tab annotate` updates material and file nodes in ISA-tab studies and assays with annotations provided as tab-separated text file.

In the annotation file header, target node types need to be indicated in ISA-tab style (i.e. "Source Name", etc.) while annotations are just named normally. Annotations for materials are automatically recorded as Characteristics, while annotations for files are recorded as Comments. Different node types can be annotated using only one annotation file, as demonstrated in the example below.

By default, if Characteristics or Comments with the same name already exist for a node type, only empty values are updated. Overwriting existing values requires confirmation (*–force-update*).

Annotations are only applied to one study and assay, since material names are not necessarily unique between the same material types of different studies or different assays (and thus, annotations couldn't be assigned unambiguously). By default the first study and assay listed in the investigation file are considered for annotation. A specific study and assay may be selected by file name (not path, just as listed in the investigation file) via *–target-study* or *–target-assay*, resp.

Example execution:

```
$ cubi-tk isa-tab annotate investigation.tsv annotation.tsv --target-study s_study.tsv
--target-assay a_assay.tsv
```

Note: investigation.tsv and annotation.tsv have to be indicated via absolute or relative paths. However, s_study.tsv and a_assay.tsv have to be indicated by name only, just as they are referenced in their corresponding investigation file.

Table 1: Annotation example tsv file

Source Name	Age	Sex	Sample Name	Volume
alpha	18	FEMALE	alpha-N1	1000
beta	27	MALE	beta-N1	1000
gamma	69	FEMALE	gamma-N1	800

4.3 More Information

Also see `cubi-tk isa-tab` CLI documentation and `cubi-tk isa-tab --help` for more information.

CHAPTER 5

Manual for `ingest-fastq`

The `cubi-tk sodar ingest-fastq` command lets you upload raw data files to SODAR. It is configured for uploading FASTQ files by default, but the parameters can be adjusted to upload any files.

The basic usage is:

```
$ cubi-tk sodar ingest-fastq SOURCE [SOURCE ...] DESTINATION
```

where each `SOURCE` is a path to a folder containing relevant files and `DESTINATION` is either an iRODS path to a *landing zone* in SODAR or the UUID of that *landing zone*.

5.1 Other file types

By default, the parameters `--src-regex` and `--remote-dir-pattern` are configured for FASTQ files, but they may be changed to upload other files as well. The two parameters have the following functions:

- `--src-regex`: a regular expression to recognize paths to raw data files to upload (the paths starting from the `SOURCE` directories).
- `--remote-dir-pattern`: a pattern specifying into which folder structure the raw data files should be uploaded. This is a file path with wildcards that are replaced by the captured content of named groups in the regular expression passed via `--src-regex`.

For example, the default `--src-regex` is

```
(.*/)?(?P<sample>.+?)(?:_(?P<lane>L[0-9]+?))?(?:_(?P<mate>R[0-9]+?))?(?:_(?P<batch>[0-
↪9]+?))?\.f(?:ast)?q\.gz
```

It can capture a variety of different FASTQ file names and has the named groups `sample`, `lane`, `mate` and `batch`. The default `--remote-dir-pattern` is

```
{sample}/{date}/{filename}
```

It contains the wildcard `{sample}`, which will be filled with the captured content of group `(?P<sample>...)`. In addition, the wildcards `{date}` and `{filename}` can always be used and will be filled with the current date and full filename (the basename of a matched file), respectively.

5.2 Mapping of file names

In some cases additional mapping of filenames is required (for example the samples should be renamed). This can be done via the parameter `--remote-dir-mapping` or short `-m`. It can be supplied several times, each time for another mapping. With each `-m MATCH REPL` a pair of a regular expression and a replacement string are specified. Internally, pythons `re.sub` command is executed on the `--remote-dir-pattern` after wildcards have been filled. Therefore, you can refer to the documentation of the re package for syntax questions.

5.3 Source files on WevDAV

If a `SOURCE` is a WebDAV url, the files will temporarily be downloaded into a directory called "./temp/". This can be adjusted with the `--tmp` option.

5.4 SODAR authentication

To use this command, which internally executes iRODS icommands, you need to authenticate with iRODS by running:

```
$ iinit
```

To be able to access the SODAR API (which is only required, if you specify a landing zone UUID instead of an iRODS path), you also need an API token. For token management for SODAR, the following docs can be used:

- https://sodar.bihealth.org/manual/ui_user_menu.html
- https://sodar.bihealth.org/manual/ui_api_tokens.html

There are three options how to supply the token. Only one is needed. The options are the following:

1. configure `~/.cubitkrc.toml`.

   ```
   [global]

   sodar_server_url = "https://sodar.bihealth.org/"
   sodar_api_token = "<your API token here>"
   ```

2. pass via command line.

   ```
   $ cubi-tk sodar ingest-fastq --sodar-url "https://sodar.bihealth.org/" --
   ↪sodar-api-token "<your API token here>"
   ```

3. set as environment variable.

   ```
   $ SODAR_API_TOKEN="<your API token here>"
   ```

5.5 More Information

Also see `cubi-tk sodar ingest-fastq` *CLI documentation* and `cubi-tk sodar ingest-fastq --help` for more information.

CHAPTER 6

Manual for `sea-snap itransfer-results`

The `cubi-tk sea-snap itransfer-results` command lets you upload results of the Seasnap pipeline to SODAR. It relies on running the `export` function of Seasnap first. This `export` function allows to select which result files of the pipeline shall be uploaded into what folder structure, which can be configured via the Seasnap config file. It outputs a `blueprint` file with file paths and commands to use for the upload. For more information see the Seasnap documentation The `itransfer-results` function parallelizes the upload of these files.

The basic usage is:

1. create blueprint

```
$ ./sea-snap mapping l export
```

2. upload to SODAR

```
$ cubi-tk sea-snap itransfer-results BLUEPRINT DESTINATION
```

where each `BLUEPRINT` is the blueprint file mentioned above (probably "SODAR_export_blueprint.txt") and `DESTINATION` is either an iRODS path to a *landing zone* in SODAR or the UUID of that *landing zone*.

6.1 SODAR authentication

To use this command, which internally executes iRODS icommands, you need to authenticate with iRODS by running:

```
$ iinit
```

To be able to access the SODAR API (which is only required, if you specify a landing zone UUID instead of an iRODS path), you also need an API token. For token management for SODAR, the following docs can be used:

- https://sodar.bihealth.org/manual/ui_user_menu.html
- https://sodar.bihealth.org/manual/ui_api_tokens.html

There are three options how to supply the token. Only one is needed. The options are the following:

1. configure `~/.cubitkrc.toml`.

```
[global]

sodar_server_url = "https://sodar.bihealth.org/"
sodar_api_token = "<your API token here>"
```

2. pass via command line.

```
$ cubi-tk sodar ingest-fastq --sodar-url "https://sodar.bihealth.org/" --
↪sodar-api-token "<your API token here>"
```

3. set as environment variable.

```
$ SODAR_API_TOKEN="<your API token here>"
```

6.2 More Information

Also see `cubi-tk sea-snap itransfer-results` *CLI documentation* and `cubi-tk sea-snap itransfer-results --help` for more information.

CHAPTER 7

Manual for `sea-snap write-sample-info`

The `cubi-tk sea-snap write-sample-info` command can be used to collect information by parsing the folder structure of raw data files (FASTQ) and meta-information (ISA-tab). It collects this information in a YAML file that will be loaded by the Seasnap pipeline.

The basic usage is:

```
$ cubi-tk sea-snap write-sample-info IN_PATH_PATTERN
```

where `IN_PATH_PATTERN` is a file path with wildcards specifying the location to FASTQ files. The wildcards are also used to extract information from the parsed paths.

By default, a file called `sample_info.yaml` will be generated in the current working directory. If this file is in the project working directory, Seasnap will load it automatically. However, you can specify another file name after `IN_PATH_PATTERN`. Then this file can be used in Seasnap e.g. like so:

```
$ ./sea-snap mapping l --config file_name='sample_info_alt.yaml'
```

Note: check and edit the auto-generated sample_info.yaml file before running the pipeline.

7.1 Path pattern and wildcards

For example, if the FASTQ files are stored in a folder structure like this:

```
input
├── sample1
│   ├── sample1_R1.fastq.gz
│   └── sample1_R2.fastq.gz
└── sample2
    ├── sample2_R1.fq
    └── sample2_R2.fq
```

Then the path pattern can look like the following:

```
$ cubi-tk sea-snap write-sample-info "input/{sample}/*_{mate,R1|R2}"
```

Keywords in braces (e.g. `{sample}`) are wildcards. It is possible to add a regular expression separated with a comma after the keyword. This is useful to restrict what part of the file path the wildcard can match (e.g. `{mate,R1|R2}` means that mate can only be `R1` or `R2`). In addition, `*` and `**` can be used to match anything that does not need to be captured with a wildcard.

Setting the `IN_PATH_PATTERN` as shown above will allow the `write-sample-info` command to extract the information that samples *sample1* and *sample2* exist and that there are *paired reads* for both of them. The extension (e.g. `fastq.gz`, `fastq` or `fq`) should be omitted and will be detected automatically.

Available wildcards are: `{sample}`, `{mate}`, `{flowcell}`, `{lane}`, `{batch}` and `{library}`. However, only **"{sample}"** is obligatory.

Note: wildcards do not match "/" and".". For further information also see the Seasnap docu.

7.2 Meta information

When working with **SODAR**, additional meta-information should be included in the sample info file. In SODAR this meta-information is stored in the form of ISA-tab files.

There are two ways to add the information from an ISA-tab assay file to the generated sample info file:

1. Load from a local ISA-tab assay file

```
$ cubi-tk sea-snap write-sample-info --isa-assay PATH/TO/a_FILE_NAME.txt IN_PATH_
↪PATTERN
```

2. Download from SODAR

```
$ cubi-tk sea-snap write-sample-info --project_uuid UUID IN_PATH_PATTERN
```

Here, `UUID` is the UUID of the respective project on SODAR.

7.3 SODAR authentication

To be able to access the SODAR API (which is only required if you download meta-data from SODAR), you also need an API token. For token management for SODAR, the following docs can be used:

- https://sodar.bihealth.org/manual/ui_user_menu.html
- https://sodar.bihealth.org/manual/ui_api_tokens.html

There are three options how to supply the token. Only one is needed. The options are the following:

1. configure `~/.cubitkrc.toml`.

```
[global]

sodar_server_url = "https://sodar.bihealth.org/"
sodar_api_token = "<your API token here>"
```

2. pass via command line.

```
$ cubi-tk sodar ingest-fastq --sodar-url "https://sodar.bihealth.org/" --
↪sodar-api-token "<your API token here>"
```

3. set as environment variable.

```
$ SODAR_API_TOKEN="<your API token here>"
```

7.4 Table format

Although this is not really necessary to run the workflow, it is possible to convert the YAML file to a table / sample sheet:

```
$ cubi-tk sea-snap write-sample-info --from-file sample_info.yaml XXX sample_info.tsv
```

And back:

```
$ cubi-tk sea-snap write-sample-info --from-file sample_info.tsv XXX sample_info.yaml
```

7.5 More Information

Also see `cubi-tk sea-snap write-sample-info` *CLI documentation* and `cubi-tk sea-snap write-sample-info --help` for more information.

CHAPTER 8

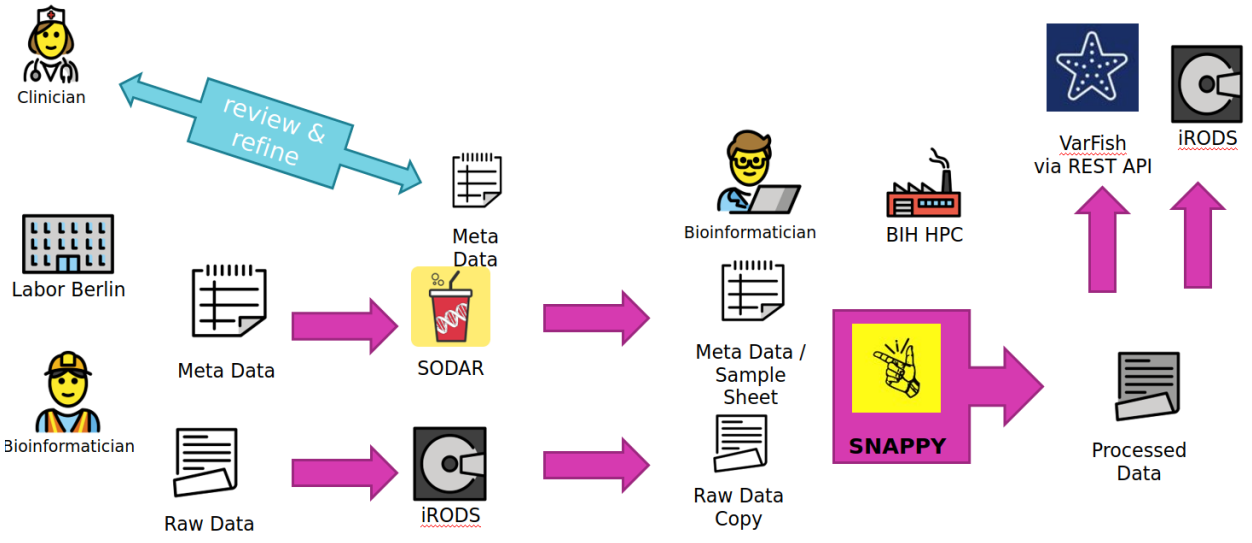
Use Case: Exomes

This section describes the cubi-tk use case for exomes that are sequenced at Labor Berlin and processed by CUBI. This section provides an outline of how cubi-tk helps in connecting

- SODAR (the CUBI system for meta and mass data storage and management),
- SNAPPY (the CUBI pipeline for the processing of DNA sequencing, including exomes),
- and VarFish (the CUBI web app for interactive analysis and annotation of variant calling results).

8.1 Overview

The overall data flow for the Translate-NAMSE use case is depicted below.



- A Labor Berlin (LB) bioinformatician uses "cubi-tk sodar add-ped" to augment the sample sheet of a SODAR project with new family members or new families alltogether. He also transfers the FASTQ read data sequences to the iRODS system that backs SODAR for file storage.
- At this stage, a Charite geneticist can review and refine the sample sheet. This mostly relates to information that is secondary for the subsequent analysis. It is assumed that the family relations updated by the bioinformatician are correct (two parents of a sample are the two parents, if father and mother are flipped, this is not important for analysis by SNAPPY).
- A CUBI Bioinformatician can now update the sample sheet for the SNAPPY pipeline using "cubi-tk snappy pull-sheets" and update a copy of the raw data sequence with "cubi-tk snappy pull-raw-data" files earlier transferred by LB.
- Once the data has been pulled from SODAR and iRODS, the CUBI bioinformatician launches the SNAPPY pipeline which processes the data on the BIH HPC. The command `cubi-tk snappy kickoff` launches the pipeline steps with their dependencies. Inspection of results is based on manual inspection of log files for now.
- Once this is complete, Manuel uses `cubi-tk snappy varfish-upload` and `cubi-tk snappy itarnsfer-{variant-calling,ngs-mapping}` to transfer the resulting BAM and VCF files into VarFish via its REST API and iRODS via landing zones (`cubi-tk sodar lz-{create,move}`).

To summarise more concisely

- LB copies data and meta data to SODAR/iRODS.
- CUBI pulls mass data and meta data form SODAR/iRODS and starts the pipeline.
- CUBI submits the resulting mass data results back into SODAR and annotated/exported variant calls into VarFish.
- The clinician can review the sample sheet independently of Manuel and Johannes.

Human interaction is required if

- The sample sheet does not sufficiently reflect reality (sample swaps)
- Files are broken and/or swapped.
- Tools terminate too early; data is not copied.
- Overall, this is not fully automated system, rather a system with heavy tool support and semi-automation.

Future improvements are

- Ask clinicians sending in samples for sex of child.
- Properly track parents as father/mother.

More Notes

- Data is processed in batches.
- Many tooling steps rely on "start processing in batch NUMBER"
- That is, everything behind NUMBER will be processed.
- Requires human-manual tracking of batch to start at (easy to seee in SODAR)

8.2 Setup

For token management for both VarFish and SODAR, the following docs can be used:

- https://sodar.bihealth.org/manual/ui_user_menu.html

- https://sodar.bihealth.org/manual/ui_api_tokens.html

1. Obtain a VarFish API token from the varfish system and configure `~/.varfishrc.toml`.

```
[global]
varfish_server_url = "https://varfish.bihealth.org/"
varfish_api_token = "<your API token here>"
```

2. Obtain a SODAR API token and configure `~/.cubitkrc.toml`.

```
[global]

sodar_server_url = "https://sodar.bihealth.org/"
sodar_api_token = "<your API token here>"
```

3. Create a new Miniconda installation if necessary.

```
host:~$ wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-
↪x86_64.sh
host:~$ bash Miniconda3-latest-Linux-x86_64.sh -b -p $HOME/miniconda3
host:~$ source $HOME/miniconda3/bin/activate
(conda) host:~$
```

4. Checkout and install VarFish CLI:

```
(conda) host:~$ git clone https://github.com/bihealth/varfish-cli.git
(conda) host:~$ cd varfish-cli
(conda) host:varfish-cli$ pip install -r requirements/base.txt
(conda) host:varfish-cli$ pip install -e .
```

5. Checkout and install CUBI-TK

```
(conda) host:~$ git clone git@cubi-gitlab.bihealth.org:CUBI/Pipelines/
↪cubi-tk.git
(conda) host:~$ cd cubi-tk
(conda) host:cubi-tk$ pip install -r requirements/base.txt
(conda) host:cubi-tk$ pip install -e .
```

8.3 SNAPPY Configuration

You have to adjust the configuration of the SNAPPY data set as follows:

- You have ot provide the `sodar_uuid` attribute. Set it to the SODAR project's UUID.
- **Data will be downloaded in the last entry of `search_paths`.**
  - If you are starting a new project then just use one entry with an appropriate value.
  - If you are moving a project to use cubi-tk then add a new entry where to download the data to.

```
# ...
data_sets:
  "<the dataset name here>:
    sodar_uuid: "<dataset uuid here>
    sodar_title: "<optional title here>
    file: "<biomedsheets file path here>.tsv"
    type: germline_variants
```

(continues on next page)

(continued from previous page)

```
    naming_scheme: only_secondary_id
    search_patterns:
    - {left: '**/*_R1.fastq.gz', right: '**/*_R2.fastq.gz'}
    - {left: '**/*_R1_*.fastq.gz', right: '**/*_R2_*.fastq.gz'}
    search_paths:
    - "<path to search data for here>"
```

Note that you will need the **/* in the pattern.

8.4 Processing Commands

The setup up to here only has to be done only once for each project/dataset. The following step will (a) fetch the meta data and raw data from SODAR/iRODS, (b) start the processing with SNAPPY, and (c) submit the results back to SODAR once SNAPPY is done.

First, you pull the meta data from SODAR with the command:

```
$ cubi-tk snappy pull-sheets
```

This will show the changes that are to be applied in unified patch format and you have to confirm by files. You can also add `--yes --dry-run` to see all pending changes at once without actually applying them or `--yes` to apply all changes.

The next step is to fetch the raw data from SODAR/iRODS. You first have to authenticate with iRODS using `init`. You then fetch the raw data, optionally only the data starting at batch number `$BATCH`. You also have to provide the project UUID `$PROJECT`. Internally, cubi-tk will use the iRODS icommands and you will be shown the commands it is about to execute.

```
$ iinit
$ cubitk snappy pull-raw-data --min-batch $BATCH $PROJECT
```

Now you could start the processing. However, it is advisable to ensure that the input FASTQ files can be linked in the `ngs_mapping` step.

```
$ cd ngs_mapping
$ snappy-snake -p $(snappy-snake -S | grep -v 'no update' | grep input_links | cut -f␣
↪1)
```

If this fails, a good starting point is removing `ngs_mapping/.snappy_path_cache`.

You can kick off the current pipeline using

```
$ cubi-tk snappy kickoff
```

After the pipeline has finished, you can create a new landing zone with the following command. This will print the landing zone properties as JSON. You will neded both the landing zone UUID (`ZONE`) and iRODS path (`$IRODS_PATH`) for now (in the future this will be simplified).

```
$ cubi-tk sodar landing-zone-create $PROJECT
```

You can then transfer the data using the following commands. You will have to specify the path to the SNAPPY sample sheet TSV as `$TSV` and the landing zone iRODS path `$IRODS_PATH`.

```
$ cubi-tk snappy itransfer-ngs-mapping --start-batch $BATCH $TSV $IRODS_PATH
$ cubi-tk snappy itransfer-variant-calling --start-batch $BATCH $TSV $IRODS_PATH
```

Finally, you can validate and move the landing zone to get the data into SODAR:

```
$ cubi-tk sodar landing-zone-move $ZONE
```

And last but not least, here is how to transfer the data into VarFish (starting at `$BATCH`).

```
$ cubi-tk snappy varfish-upload --min-batch $BATCH $PROJECT
```

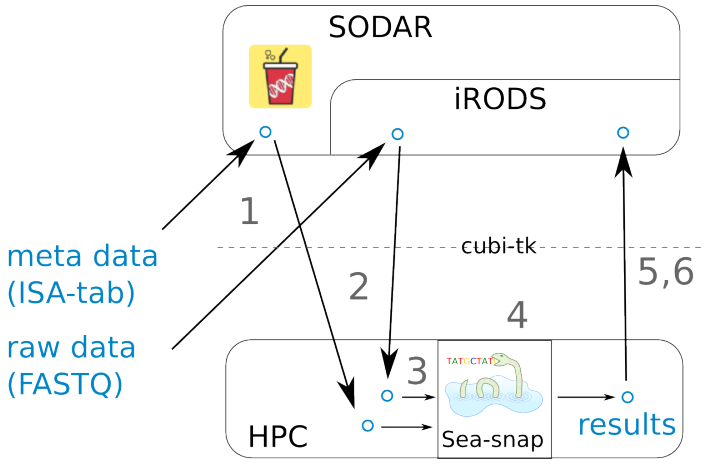
CHAPTER 9

Use Case: Single Cell

This section describes the cubi-tk use case for the analysis of single cell data. It provides an outline of how cubi-tk helps in connecting

- Sea-Snap (the CUBI pipeline for the processing of RNA sequencing, including scRNA-seq),
- SODAR (the CUBI system for meta and mass data storage and management).

9.1 Overview



1 FASTQ and ISA-tab files are uploaded to SODAR.

- ISA-tab files can be created with the help of `cubi-tk isa-tpl isatab-single_cell`.
- FASTQ files can be uploaded with the help of `cubi-tk sodar ingest-fastq`

2 FASTQ and ISA-tab files are pulled from SODAR.

- FASTQ files can be downloaded using `cubi-tk sodar pull-raw-data` or iRods icommands.

- ISA-tab files can be downloaded using `cubi-tk sea-snap pull-isa`.

3 A results folder is created on the HPC cluster and the config files are edited. A sample info file is created.

- A results folder can be created with `cubi-tk sea-snap working-dir`.
- The sample_info.yaml file can be created with `cubi-tk sea-snap write-sample-info`. This combines information from the parsed FASTQ folder structure and ISA-tab meta information.

4 Running the Sea-snap pipeline.

- This is done as usual via `./sea-snap sc --slurm c`.

5 The results are uploaded to SODAR.

- Create a landing zone on SODAR with `cubi-tk sodar lz-create`.
- Create a blueprint of which files to upload with `./sea-snap sc l export`.
- Upload the results using the blueprint and `cubi-tk itransfer-results`.

6 Check whether all files have been uploaded to SODAR correctly.

- This can be done via `cubi-tk sea-snap check-irods`.

9.2 Setup

For token management for SODAR, the following docs can be used:

- https://sodar.bihealth.org/manual/ui_user_menu.html
- https://sodar.bihealth.org/manual/ui_api_tokens.html

1. Obtain a SODAR API token and configure `~/.cubitkrc.toml`.

```
[global]

sodar_server_url = "https://sodar.bihealth.org/"
sodar_api_token = "<your API token here>"
```

2. Create a new Miniconda installation if necessary.

```
host:~$ wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-
↪x86_64.sh
host:~$ bash Miniconda3-latest-Linux-x86_64.sh -b -p $HOME/miniconda3
host:~$ source $HOME/miniconda3/bin/activate
(conda) host:~$
```

3. Checkout and install CUBI-TK

```
(conda) host:~$ git clone git@cubi-gitlab.bihealth.org:CUBI/Pipelines/
↪cubi-tk.git
(conda) host:~$ cd cubi-tk
(conda) host:cubi-tk$ pip install -r requirements/base.txt
(conda) host:cubi-tk$ pip install -e .
```

9.3 Processing Commands

Hint: Also see the Seasnap single cell pipeline documentation here.

First, you can pull the meta data from SODAR with the command:

```
$ cubi-tk sea-snap pull-isa <project_uuid>
```

This will create a folder with ISA-tab files. Alternatively, you can omit this step and automatically pull the files later.

The next step is to fetch the raw data from SODAR/iRODS. You first have to authenticate with iRODS using `iinit`. Internally, cubi-tk will use the iRODS icommands and you will be shown the commands it is about to execute.

```
$ iinit
$ cubi-tk sodar pull-raw-data <project_uuid>
```

Create a working directory for the project results:

```
$ cubi-tk sea-snap working-dir <path_to_seasnap_pipeline>
```

This will also copy relevant files and a config template into the new directory. Edit the config files to adjust the pipeline execution to your needs.

Create a sample info file. This is equivalent to a sample sheet and summarizes information about the samples in yaml format. A path pattern to the downloaded FASTQ files is needed, see Sea-snap doku: https://cubi-gitlab.bihealth.org/CUBI/Pipelines/sea-snap/blob/master/documentation/prepare_input.md#fastq-files-folder-structure

```
$ cubi-tk sea-snap write-sample-info --isa-assay <path_to_assay_file> <path_pattern_
↪to_fastq>
```

This combines information from both the FASTQ folder structure (given via path pattern) and the ISA-tab meta data (given via ISA-assay file). If ISA-tab files have not been downloaded yet, you can use the option `--project-uuid <project_uuid>` instead of `--isa-assay` to download them on-the-fly.

Now you can start the processing. Run the Sea-snap pipeline as usual:

```
$ ./sea-snap sc --slurm c <any snakemake options>
$ ./sea-snap sc --slurm c export
```

After the pipeline has finished, you can create a new landing zone with the following command. This will print the landing zone properties as JSON. You will need the landing zone UUID (`ZONE`) in the next step.

```
$ cubi-tk sodar landing-zone-create <project_uuid>
```

You can then transfer the data using the following commands. You will have to specify the blueprint file generated by the export rule of sea-snap.

```
$ cubi-tk sea-snap itransfer-results <blueprint_file> <landing_zone_uuid>
```

Finally, you can validate and move the landing zone to get the data into SODAR:

```
$ cubi-tk sodar landing-zone-move <landing_zone_uuid>
```

You may check, whether everything was uploaded correctly using the following command:

```
$ cubi-tk sea-snap check-irods <path_to_local_results_folder> <irods_path_to_results_
↪on_sodar>
```

CHAPTER 10

Credits

- Eudes Bargos
- Johannes Helmuth
- Manuel Holtgrewe
- Patrick Pett

CHAPTER 11

HISTORY

History

v0.3.0

- Moving SODAR REST API calls to package *sodar-cli*.
- Switching to Github actions for CI tests.
- More templates for *cubi-tk isa-tpl*.
- Improvements and fixes to *cubi-tk sea-snap*.
- Adding *isa-tab add-ped* command.
- More tools for *cubi-tk sodar*.
- Temporarily working around SODAR REST API not returning sodar_uuid where we expect it to.
- Using library_ name as an alternative to folder_name.
- Adding *cubi-tk isa-tab annotate* command.
- Various small fixes and adjustments.

v0.2.0

- Adjusting package meta data in *setup.py*.
- Fixing documentation bulding bug.
- Documentation is now built during testing.
- Adding *cubi-tk snappy pull-sheet*.
- **Converting *snappy-transfer_utils*, adding *cubi-tk snappy ...***
  - *itransfer-raw-data*
  - *itransfer-ngs-mapping*
  - *itransfer-variant-calling*
- Adding *mypy* checks to CI.

- Adding *–dry-run* and *–show-diff* arguments to *cubi-tk snappy pull-sheet*.
- Adding *cubi-tk snake check* command.
- Adding *cubi-tk isa-tab validate* command.
- Adding *cubi-tk isa-tab resolve-hpo* command.
- Adding *cubi-tk sodar download-sheet* command.
- Adding *cubi-tk snappy kickoff* command.
- Adding *cubi-tk org-raw {check,organize}* command.
- *cubi-tk snappy pull-sheet* is a bit more interactive.
- Adding *cubi-tk sea-snap pull-isa* command.
- Adding *cubi-tk sea-snap write-sample-info* command.
- Adding *cubi-tk sea-snap itransfer-mapping-results* command.
- Adding more tools for interacting with SODAR.
- Rebranding to *cubi-tk* / CUBI Toolkit

v0.1.0

- Bootstrapping *cubi-tk* with ISA-tab templating via *cubi-tk isa-tpl <tpl>*.

CHAPTER 12

License

You can find the License of AltamISA below.

```
MIT License

Copyright (c) 2020-2021, Berlin Institute of Health

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.
```

Python Module Index

C

cubi_tk.isa_tab, 43
cubi_tk.isa_tpl, 41

Index

C

cubi_tk.isa_tab (*module*), 43
cubi_tk.isa_tpl (*module*), 41