Example STAR Product Description

Product Description Checklist:

All product descriptions must include the following items[right click on box and select checkmark when completed]:

- ❏ **Brief** 1 paragraph description of app/workflow (User Story)
  A scientific justification/backstory is undesirable in this document and does nothing to inform the developers about implementation
- ❏ Links to source for downloading the tool(s) to be wrapped
- ❏ A folder containing an end to end run of the tool(s) being wrapped. The developer should be able to fully replicate an example run to validate that they have complete specifications, as well validating that the wrapped app produces identical results. This should be stored in an anonymously FTP accessible directory, or in a KBase Box folder
  - ❏ Example source files used as inputs
  - ❏ A script that runs the tool against the input files, with all required command line arguments, as well as the commands to generate desired reports against the output files
  - ❏ The relevant output files from running B - this includes the output from the tool, as well as any reports that are created to assess the quality of the output files
- ❏ A clear description of how the input and output files map into existing KBase data types, or else a description of any new data types that need to be developed.
  - ❏ Any new data types needs to be explained and the relationship of the new types to existing data types must be documented
    New data types cannot simply be wrappers for output files
  - ❏ Appropriate file types that can be uploaded/downloaded into/out of the new type must be documented along with example files for testing upload/download
- ❏ A diagram documenting the input data types and the output data types should be included, this is especially required for complex, multi-step workflows
- ❏ Mockups of the input and output must be included that show the required and optional fields, as well as the what the output report should look like

User Stories

These user stories describe the actual task that a user would like to accomplish with RNASeq express app - as a result they provide guidance for requirements for the Minimal Viable Product that can be inferred even if they are not fully spelled out in the product description.

1. As a user I would like to upload my RNA Seq reads files into KBase for analysis. This step is included for completeness and to verify the end to end functionality. It should already be implemented.

2. As a user I would like to be able to easily generate an expression matrix for the RNA Seq reads or set of reads that I have uploaded to KBase.
3. As a user I would like to download the expression matrix generated from the RNA Seq analysis for offline analysis. This step is included for completeness and to verify the end to end functionality. It should already be implemented.

Sources for Building Star App

https://github.com/alexdobin/STAR

Files from End to End run of STAR

A Box folder with data and a script for running the STAR executable against the source data can be found at https://app.box.com/folder/32647477743

It includes the original source data for reads and for the reference genome, a script “run_star.sh” which performs the alignments and genecounts, as well as creating an output report for the alignments using qualimap. The genecounts output generated by the --quantMode option to STAR, and then collected into a single file by the extract_expression.py script. The counts would need to be normalized when creating the differential expression matrix.

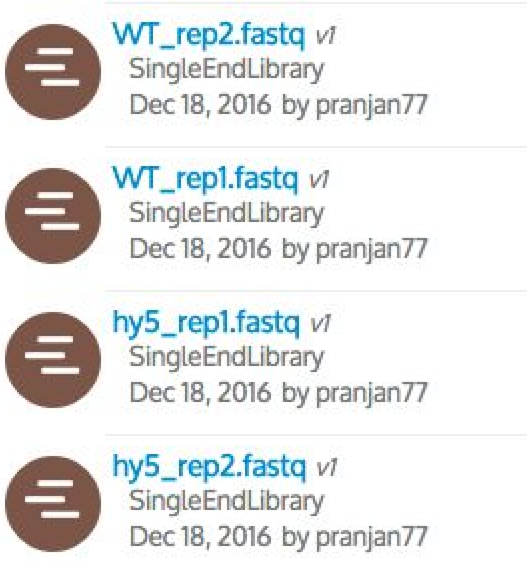
There is also a tarball containing a sample qualimap report run against the input data.

The source files for the reads and reference genome are from the example data provided by Sunita.

Example Data for Testing

The typical data for testing are reads based on Arabidopsis thaliana. This Jira ticket from Sunita describes the sources for the reads that are used for the RNA Seq examples:
https://kbase-jira.atlassian.net/browse/KBASE-4939

When using a narrative, to avoid downloading and uploading the reads files, copy the SingleEndLibraries from the RNASeq test narrative:
https://appdev.kbase.us/narrative/ws.2489.obj.1



WT_rep2.fastq v1
SingleEndLibrary
Dec 18, 2016 by pranjan77

WT_rep1.fastq v1
SingleEndLibrary
Dec 18, 2016 by pranjan77

hy5_rep1.fastq v1
SingleEndLibrary
Dec 18, 2016 by pranjan77

hy5_rep2.fastq v1
SingleEndLibrary
Dec 18, 2016 by pranjan77

For the genome to use for alignment, use an Arabidopsis Thaliana genome such as Athaliana_PhytozomeV11_TAIR10 (available in CI).

Required Data Types

For the most part, the required data types already exist within KBase for this product description, however for completeness and as a reference for PDs that require a new data type, we include a description of the input SampleSet and output data type. For the product description, the data definition need not be formal, however providing a high level description of the requirements will allow developers to fill in the details. Ideally existing KBase-wide datatypes should be used to avoid silo-ing data into app specific types. In addition, any output from an app which is used as an input to another app within KBase should be a formal type - and not simply the raw output files from running the program.

The simplest methods is to simply providing a reference to an existing viewer or reporting tool - this should determine what needs to be in the output object.

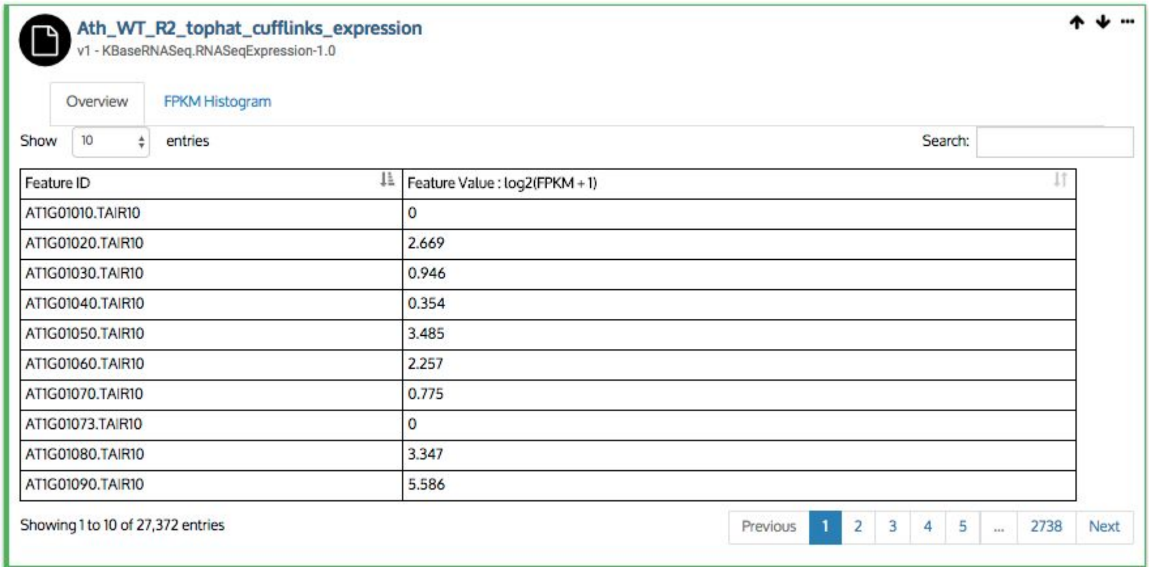
Definition RNASeqSampleSet

An RNASeq SampleSet should have the following fields

- A name for the SampleSet
- A textual description of the SampleSet
- The domain (euk, prok) for the samples
- A list of RNASeq reads in the set containing pairs of
    - a library ID for either a single or paired end library read library
    - A textual treatment label for the library above
- The type of the read library

Definition of the Expression Matrix

The output expression matrix should be of the type RNASeqExpression and be otherwise compatible with the outputs from the assembly tools, StringTie and Cufflinks. Here is the example viewer from an existing RNASeqExpression object:

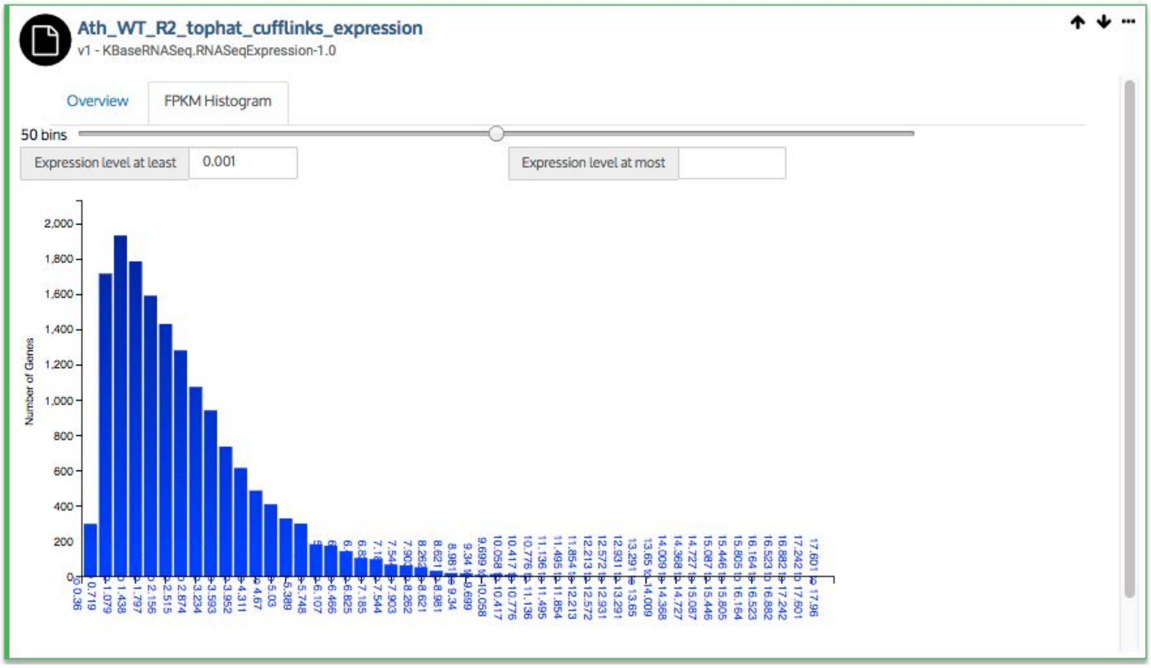


Ath_WT_R2_tophat_cufflinks_expression
v1 - KBaseRNASeq.RNASeqExpression-1.0

Show 10 entries Search:

Feature ID	Feature Value : log2(FPKM + 1)
AT1G01010.TAIR10	0
AT1G01020.TAIR10	2.669
AT1G01030.TAIR10	0.946
AT1G01040.TAIR10	0.354
AT1G01050.TAIR10	3.485
AT1G01060.TAIR10	2.257
AT1G01070.TAIR10	0.775
AT1G01073.TAIR10	0
AT1G01080.TAIR10	3.347
AT1G01090.TAIR10	5.586

Showing 1 to 10 of 27,372 entries



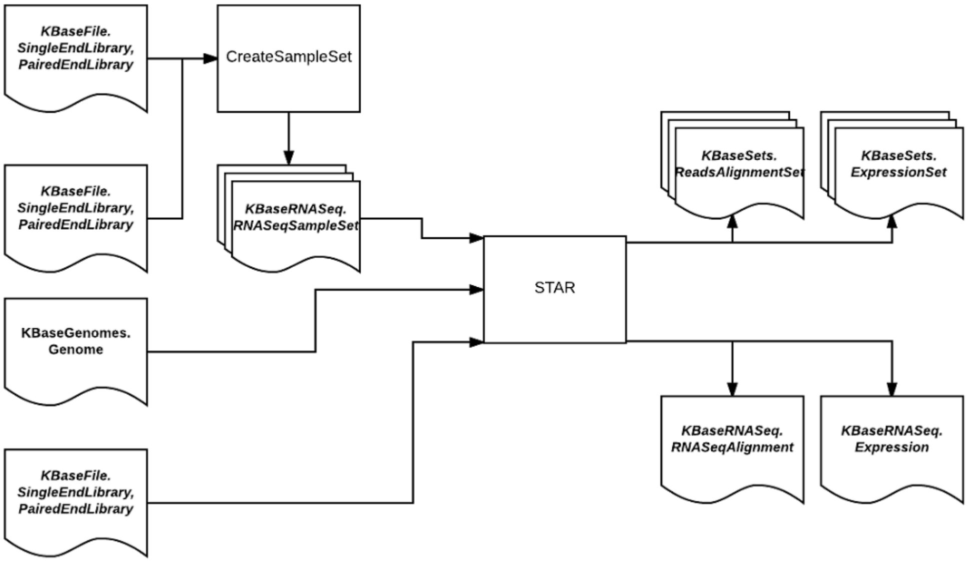
Ath_WT_R2_tophat_cufflinks_expression
v1 - KBaseRNASeq.RNASeqExpression-1.0

50 bins
Expression level at least 0.001
Expression level at most

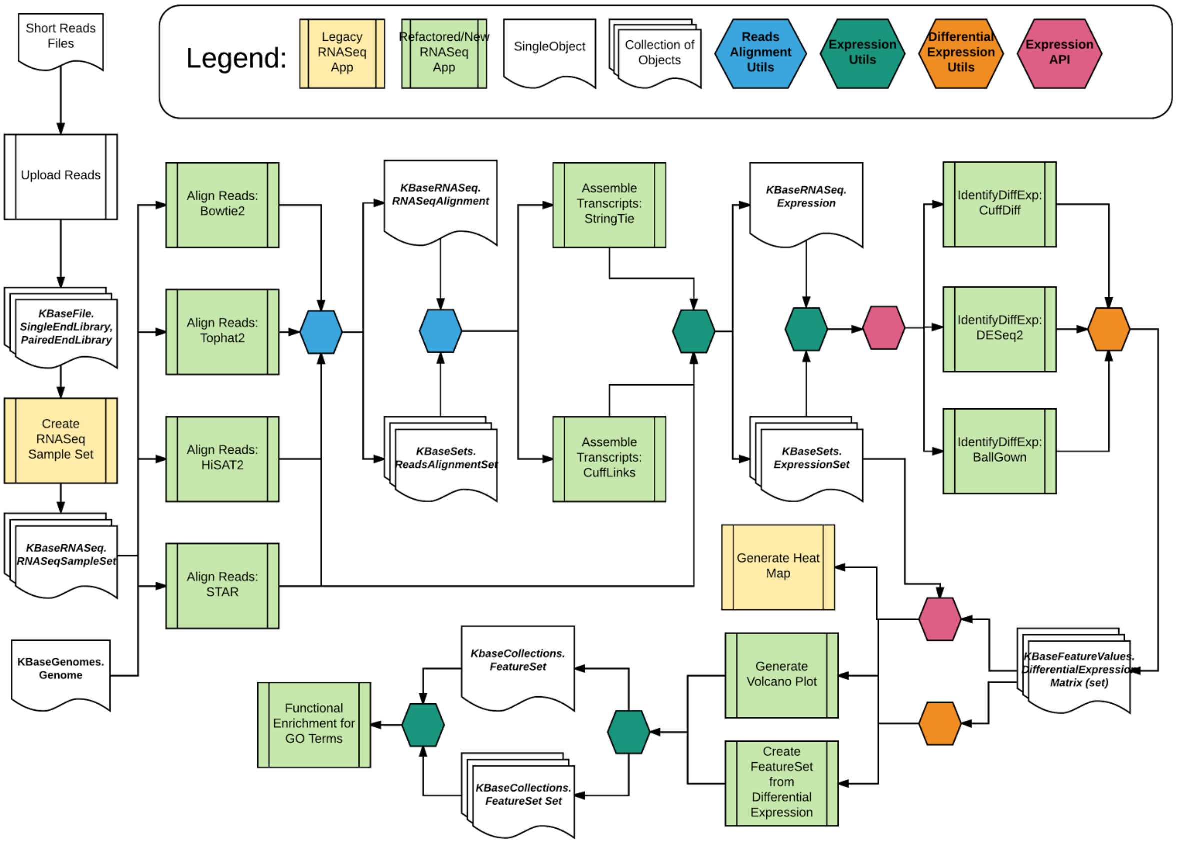
Definition of downloaded object

RNASeqExpression objects have existing downloaders, and they should be used.

Diagram of inputs and outputs for App Workflow



Here is the STAR app in the context of the broader RNASeq workflow, showing how STAR is able to bypass the assembly step and directly output files that can be used to generate Expression Matrices:

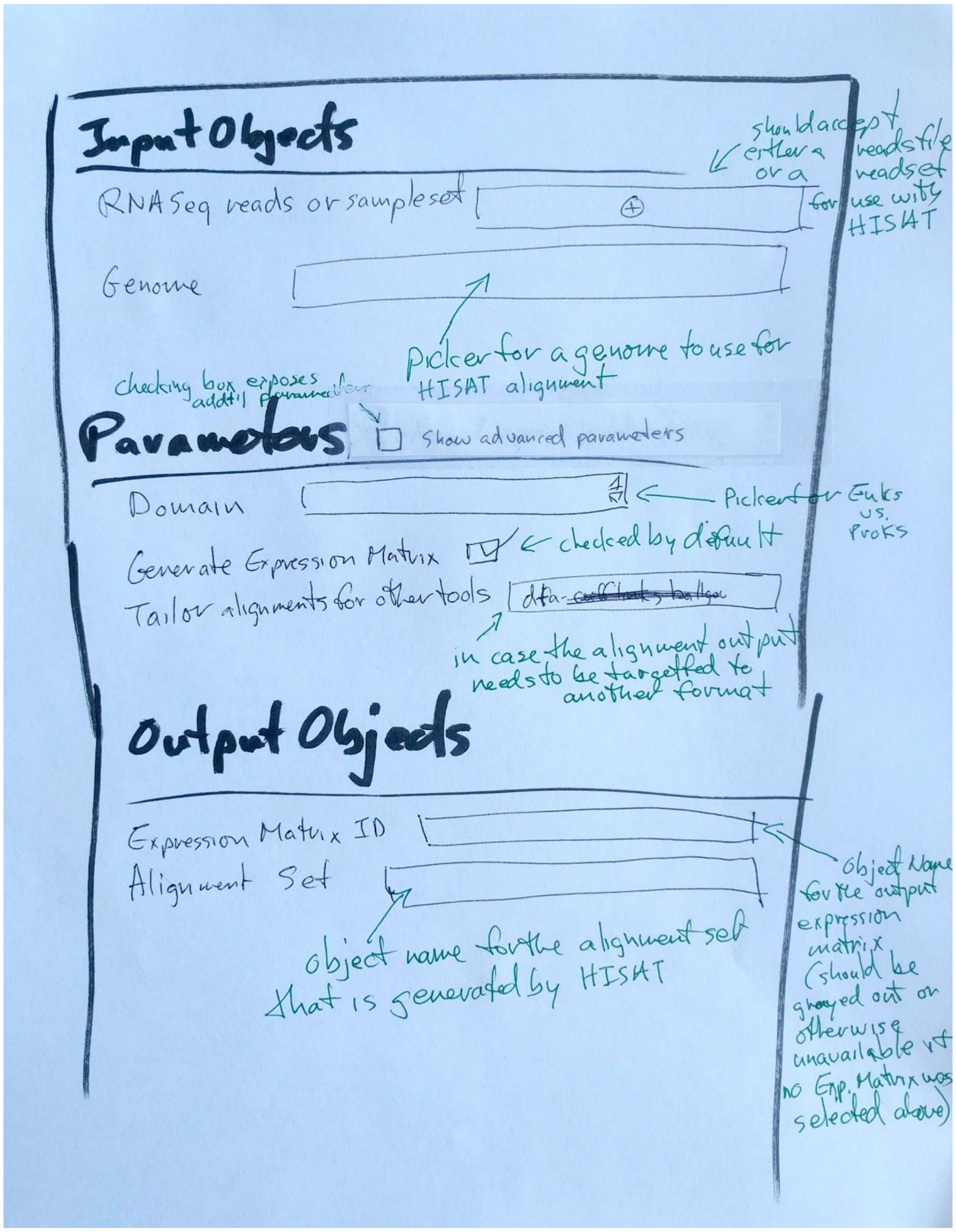


Mockups

The following mockups will concentrate on step 2 of the User Stories - the upload and download steps should already exist within KBase. The underlying tools are the HISAT2/StringTie/Ballgown tools which should already be available in the current KBaseRNASeq module.

App Cell Input

The following are 3 progressively more reified versions of the AppCell mockup, ranging from a hand drawn sketch of the app cell with notes, to a graphical mockup with annotations, to a spec file that can be used to generate the mock UI within the app:


Input Objects
RNASeq reads or sampleset
Genome
Parameters
□ Show advanced parameters
Domain
Generate Expression Matrix ☑
Tailor alignments for other tools
dfa-~~cufflinks, ballgow~~
Output Objects
Expression Matrix ID
Alignment Set

should accept either a or a readsfile readset for use with HISAT

picker for a genome to use for HISAT alignment

checking box exposes addt'l parameters

Picker for Euks vs. Proks

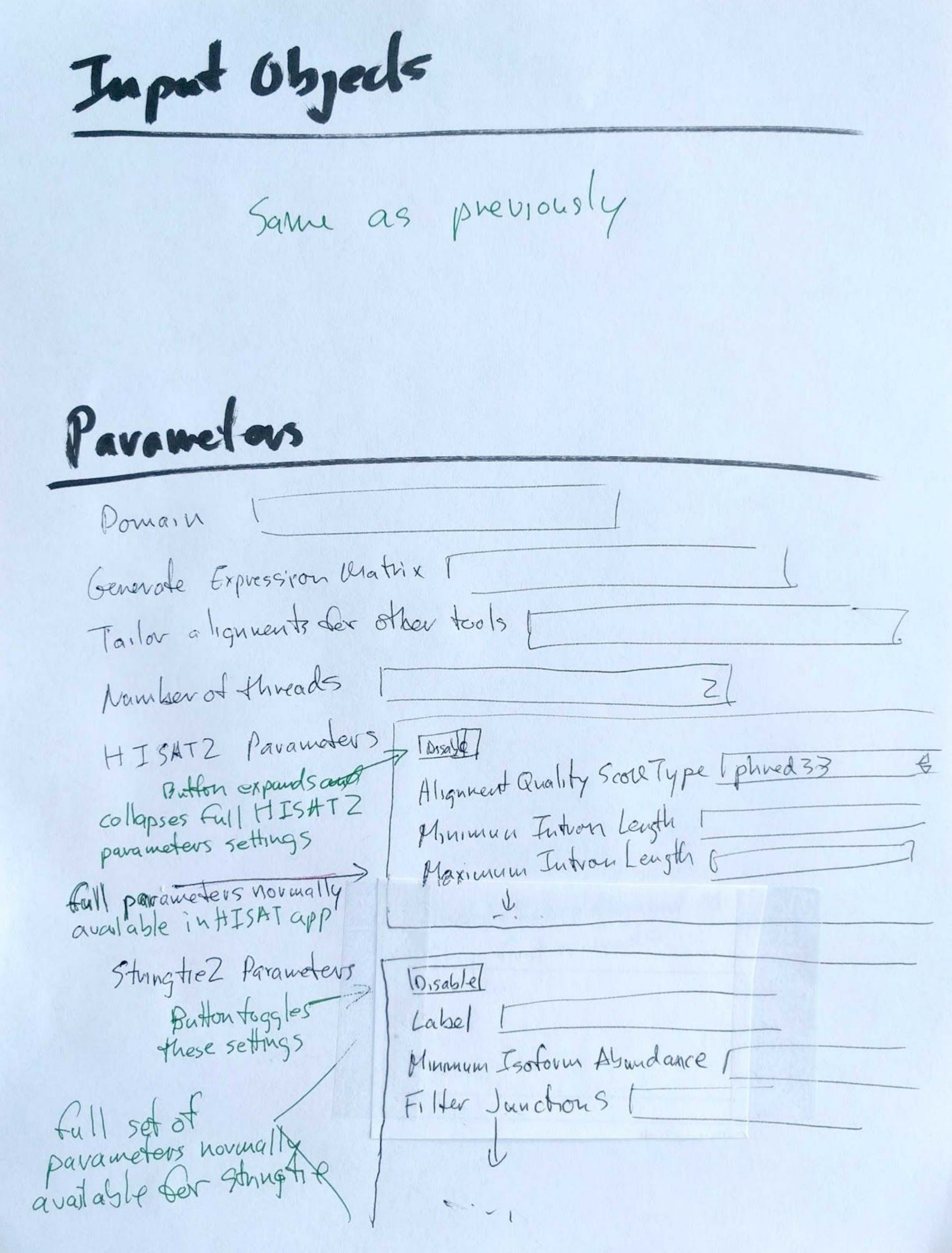
checked by default

in case the alignment output needs to be targetted to another format

Object Name for the output expression matrix (should be grayed out or otherwise unavailable if no Exp. Matrix was selected above)

object name for the alignment set that is generated by HISAT

App Cell with collapsed Advanced parameters



Input Objects

Same as previously

Parameters

Domain

Generate Expression Matrix

Tailor alignments for other tools

Number of threads 2

HISAT2 Paramaters

Button expands ~~and~~ collapses full HISAT2 parameters settings

Disable

Alignment Quality Score Type phred33

Minimum Intron Length

Maximum Intron Length

full parameters normally available in HISAT app

Stringtie2 Parameters

Button toggles these settings

Disable

Label

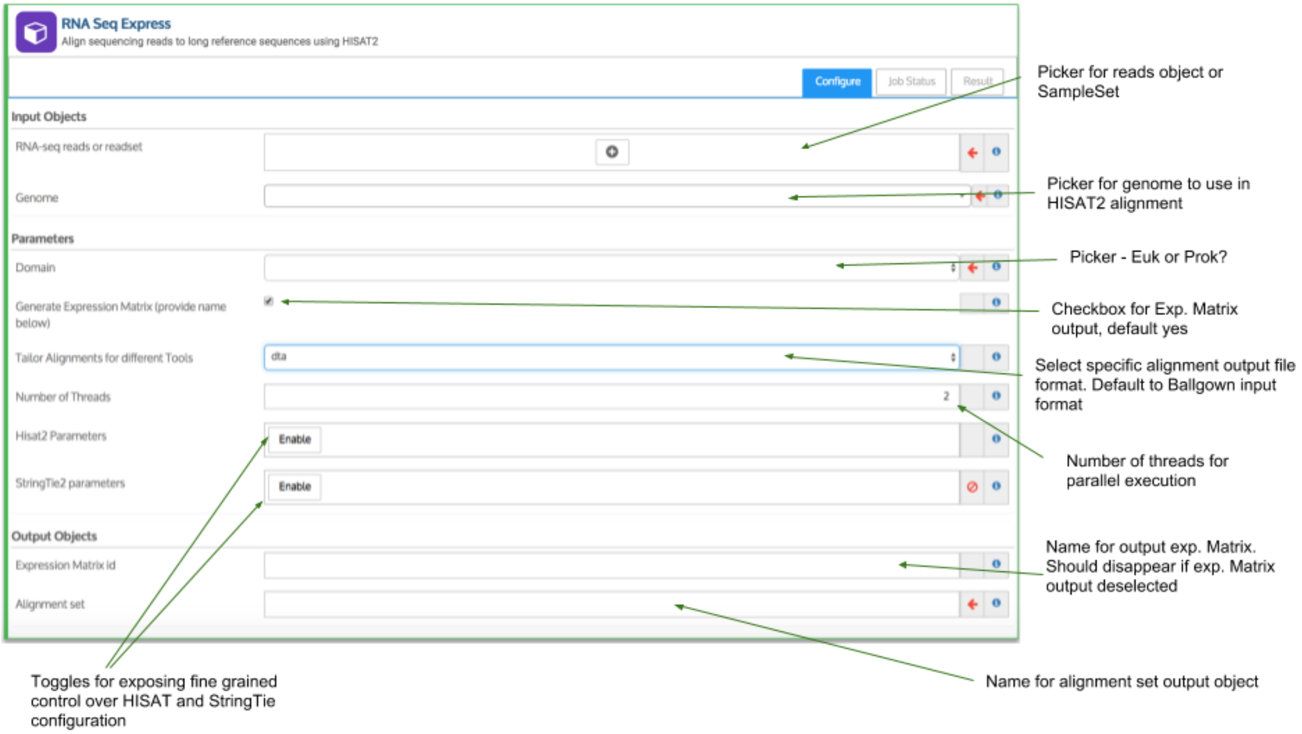
Minimum Isoform Abundance

Filter Junctions

full set of parameters normally available for stringtie

App Cell with Advanced parameters Expanded

These are examples of the mockups done on the computer (we are using an existing mockup, but any drawing tool would work). The labels for each of the fields should serve as a guideline for a tooltip that appears for each field.



RNA Seq Express
Align sequencing reads to long reference sequences using HISAT2

Input Objects

RNA-seq reads or readset

Genome

Parameters

Domain

Generate Expression Matrix (provide name below) ☑

Tailor Alignments for different Tools: dta

Number of Threads: 2

Hisat2 Parameters

StringTie2 parameters

Output Objects

Expression Matrix id

Alignment set

Picker for reads object or SampleSet

Picker for genome to use in HISAT2 alignment

Picker - Euk or Prok?

Checkbox for Exp. Matrix output, default yes

Select specific alignment output file format. Default to Ballgown input format

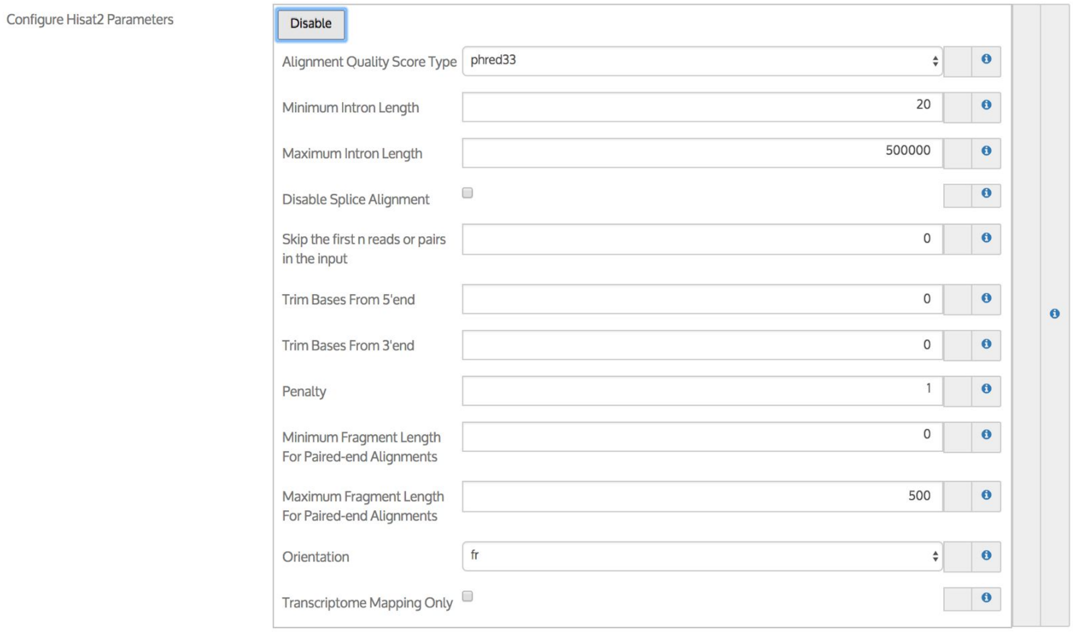
Number of threads for parallel execution

Name for output exp. Matrix. Should disappear if exp. Matrix output deselected

Toggles for exposing fine grained control over HISAT and StringTie configuration

Name for alignment set output object

Mockup with notes about fields (detailed parameter configurations hidden)



Configure Hisat2 Parameters

Alignment Quality Score Type: phred33

Minimum Intron Length: 20

Maximum Intron Length: 500000

Disable Splice Alignment ☐

Skip the first n reads or pairs in the input: 0

Trim Bases From 5'end: 0

Trim Bases From 3'end: 0

Penalty: 1

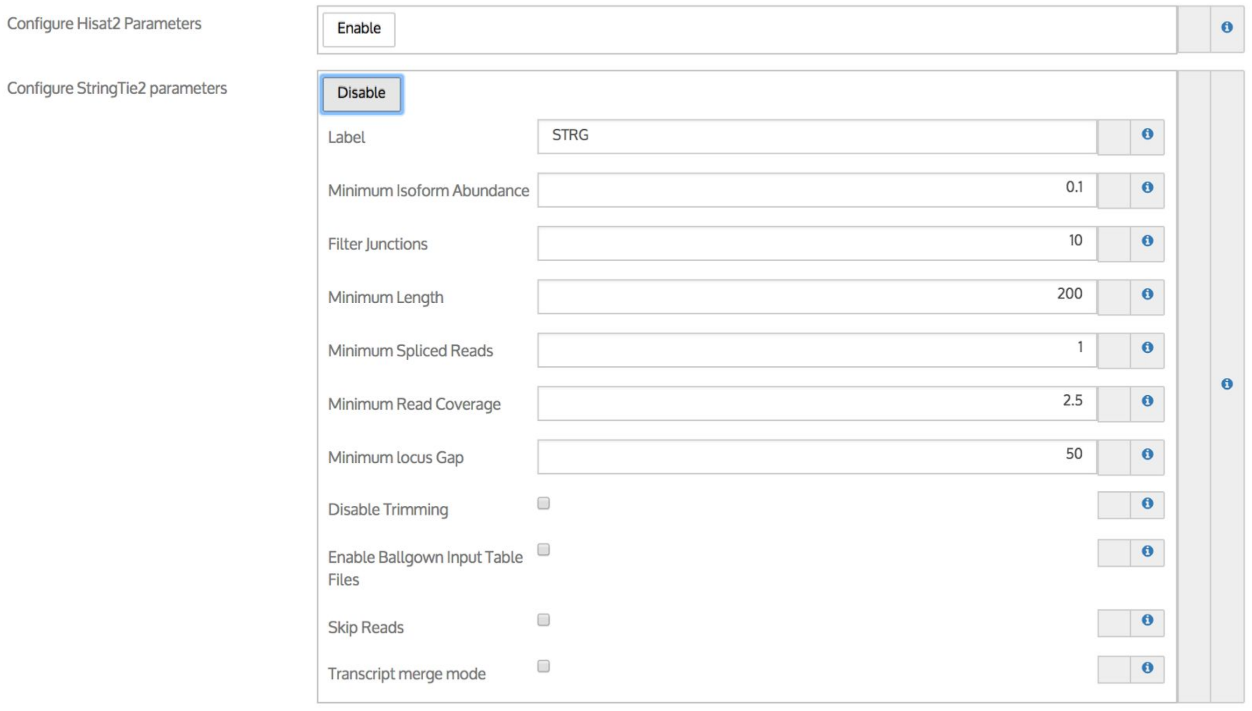
Minimum Fragment Length For Paired-end Alignments: 0

Maximum Fragment Length For Paired-end Alignments: 500

Orientation: fr

Transcriptome Mapping Only ☐

Close up of the HISAT2 Parameters section with HiSAT2 Params exposed (identical to normal HISAT2 app cell options)



Configure Hisat2 Parameters

Configure StringTie2 parameters

Label: STRG

Minimum Isoform Abundance: 0.1

Filter Junctions: 10

Minimum Length: 200

Minimum Spliced Reads: 1

Minimum Read Coverage: 2.5

Minimum locus Gap: 50

Disable Trimming □

Enable Ballgown Input Table Files □

Skip Reads □

Transcript merge mode □

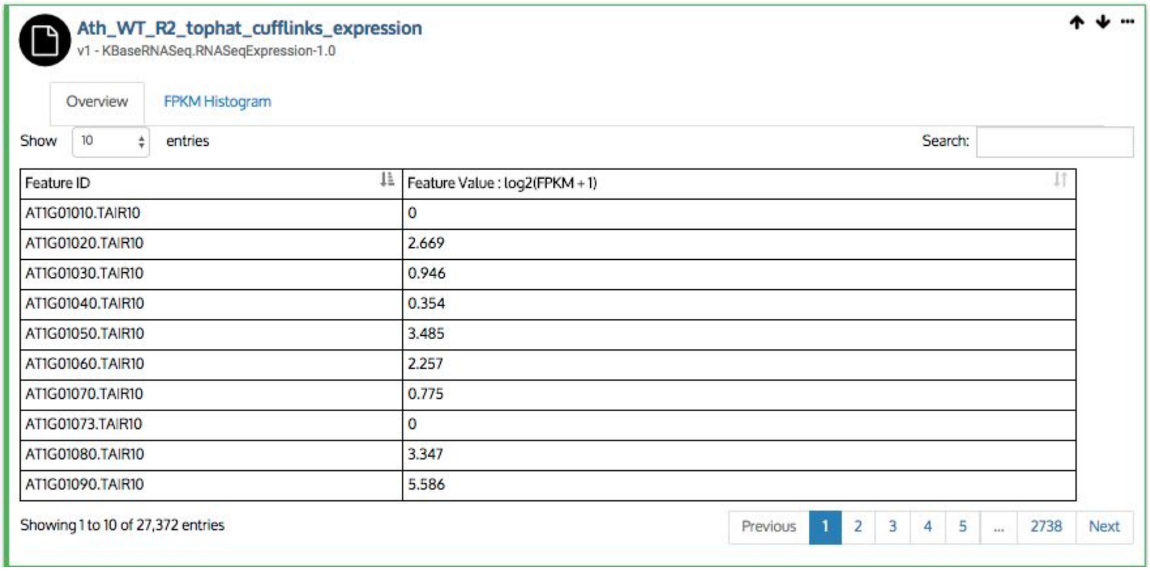
Example of StringTie2 parameters exposed (and Hisat2 collapsed). Should be identical to StringTie2 app cell parameters.

Sample Repo with UI: https://github.com/sychan/KBaseRNASeq/tree/express/ui/narrative/methods/align_reads_and_assemble_transcripts_using_hisat2_and_stringtie

(note that the specs in that file still need some updating to match the UI displayed in mockups above.)

App Cell Output

The output from running the app should be a table of the objects generated as well as a viewer for the expression matrix, see the screenshot of a Differential Expression Matrix typically output from cufflinks:



Ath_WT_R2_tophat_cufflinks_expression
v1 - KBaseRNASeq.RNASeqExpression-1.0
Overview
FPKM Histogram
Show 10 entries
Search:

Feature ID	Feature Value : log2(FPKM + 1)
AT1G01010.TAIR10	0
AT1G01020.TAIR10	2.669
AT1G01030.TAIR10	0.946
AT1G01040.TAIR10	0.354
AT1G01050.TAIR10	3.485
AT1G01060.TAIR10	2.257
AT1G01070.TAIR10	0.775
AT1G01073.TAIR10	0
AT1G01080.TAIR10	3.347
AT1G01090.TAIR10	5.586

Showing 1 to 10 of 27,372 entries

Previous 1 2 3 4 5 ... 2738 Next