ACIB PGDB Toolbox 1.0
User Guide

Peter M. Krempl
Gerhard G. Thallinger

Core Facility Bioinformatics
Austrian Centre of Industrial Biotechnology

c/o Institute for Genomics and Bioinformatics
Graz University of Technology
Petersgasse 14 / V
8010 Graz
Austria

June 4, 2012

Contents

Chapter 1

Gene Expression Data Integration

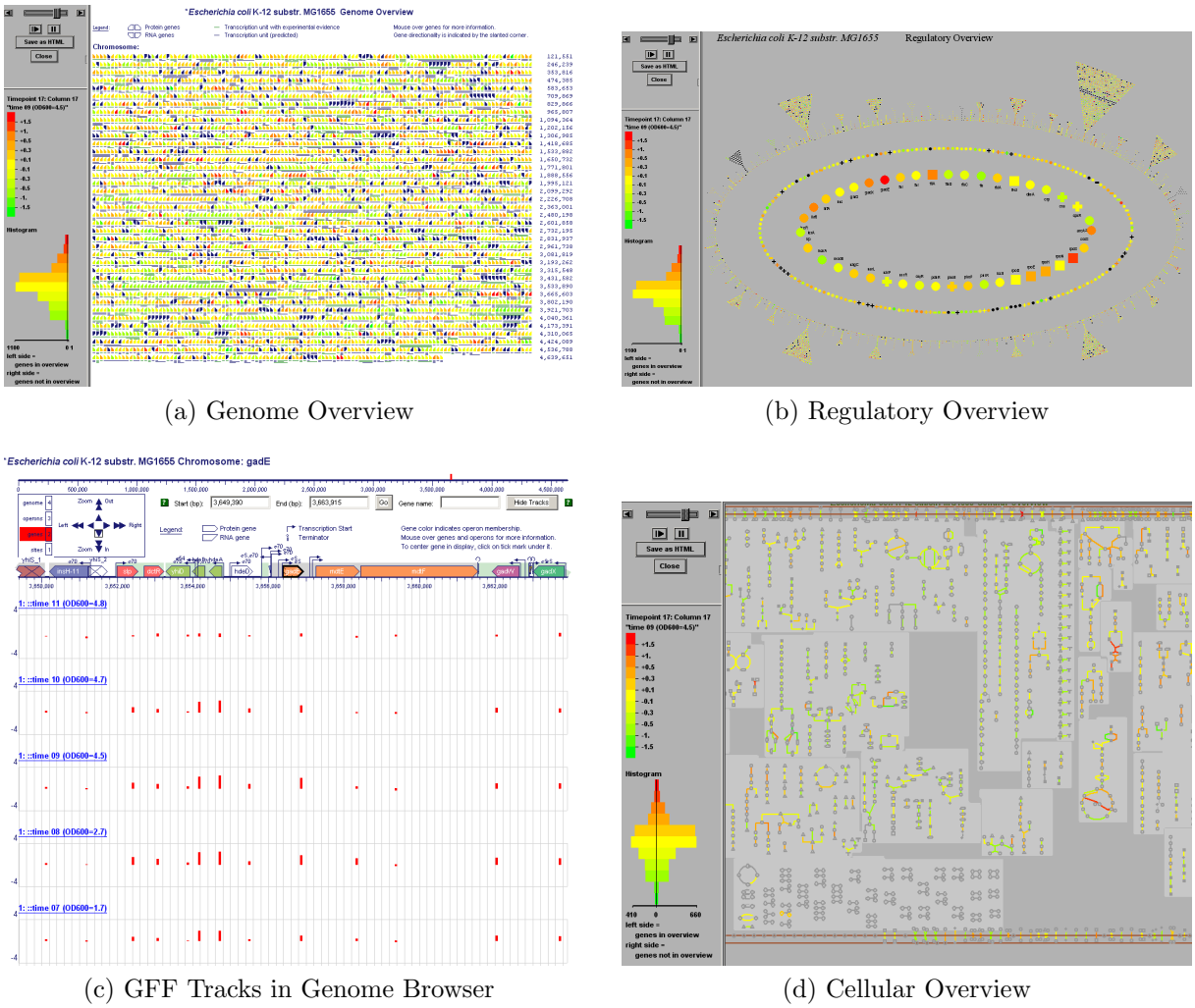
Pathway Tools offers a number of tools to facilitate the interpretation of high-throughput gene expression experiments in a global metabolic and genomic context. Its Cellular, Regulatory and Genome Overviews can illustrate experimental results by color coding gene expression levels and creating animations of time series experiments. The Genome Browser itself allows to create additional annotation tracks from GFF files, offering different modes to display the scores stored in such files.

The ACIB Toolbox provides tools to facilitate the creation of input files for these data visualization features. Starting from primary experimental result data files - either created in the own lab, or retrieved from a public database like ArrayExpress -, users can

- import array designs, from different standard file formats,
- map the probes of these arrays to their genomes / PGDBs, and finally
- convert gene expression experiment data into formats supported by Pathway Tools' data visualization tools.

Microarray probes can be imported from tab-delimited text files (e.g. GAL or ADF files) using a generic parser that prompts the user to select proper probe ID and sequence columns. Array designs and probes are stored in a central subdirectory of the Pathway Tools working directory. For each array design, tables of probe IDs and sequences, metadata about the design as well as input files for sequence alignment are stored. Probes are mapped to target genomes/PGDBs by BLAST, followed by extraction of strong hits. Mapping results are stored in GFF2 format, which allows displaying probe target annotation as additional track in Pathway Tools' genome browser.

To map expression data to the genome, tab-delimited data files are merged with probe-target-mappings mentioned above. A configuration file defines how expression data is organized into different tracks; time series, clusters and optional selection/highlighting of targets on either strand are supported. Results are stored in two formats: GFF2 to be visualized in the genome browser, and tab-delimited text files that link expression levels to PGDB-internal gene IDs for visualization in Pathway Tools genome, cellular and regulatory overview diagrams (see Figure 1.1).



(a) Genome Overview

(b) Regulatory Overview

(c) GFF Tracks in Genome Browser

(d) Cellular Overview

Figure 1.1: Visualization of Gene Expression Data in Pathway Tools, using input data files generated by ACIB Toolbox
Experimental data: Reference Design time-course, *E. coli* (Takahashi et al., 2011)
ArrayExpress Accession E-GEOD-6033
http://www.ebi.ac.uk/arrayexpress/experiments/E-GEOD-6033

The experiment data used to generate the input files for all data visualizations shown in Figure 1.1, as well as some auxiliary files, are available as example data. For a detailed description on where the example files are located on your system and how they might be used, please see Appendix B.1.

1.1 Probe and Array Design Administration

1.1.1 Import and Update of Probe Libraries and Array Designs

Probe Libraries / Array Designs can be imported from 3 different text file formats:

1. simple tab-delimited text (with header row)
2. ArrayExpress Array Design Format (ADF)
3. GenePix Array List (GAL)

In all cases, the input file must at least provide columns specifying the **probe sequence** as well as a **unique identifier** for each probe. Probes - i.e. probe IDs - may occur multiple times within the input file, as long as probe information is the same at each occurrence. In case a probe ID happens to be associated with differing probe data lines, the information of the last such line occurring in the file will be used. Simple tab-delimited text files must have a fixed number of columns, the first row must contain the column headers; comment lines are not supported.

To start probe import, select
ACIB-Toolbox
└ ***Gene Expression Data Integration***
└ ***Import or update Probe Library / Array Design***
from the menu. A dialog will pop up to aks for the input format and file (Figure 1.2).

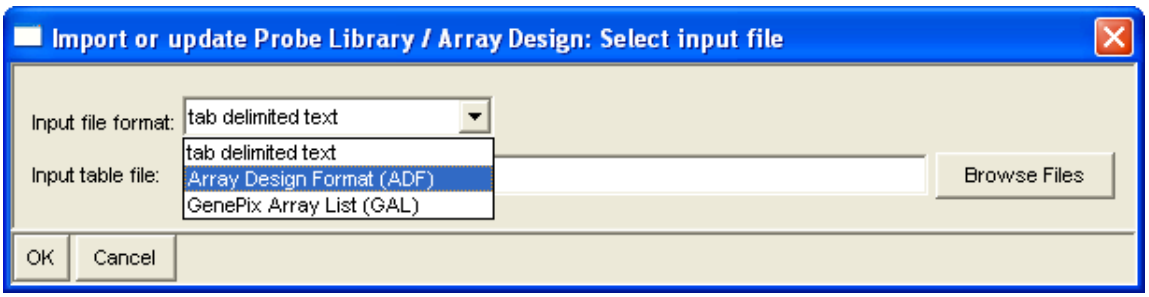


Figure 1.2: Import Probes: input file selection dialog

Once the input file is loaded, another dialog will prompt for the selection or creation of a Probe Library (Figure 1.3). Depending on the input file format, the import function may be able to determine Probe Library ID and name from the input file's metadata. In this case these values are also shown in the dialog, and the respective Probe Library - if already present - will be preselected in the dropdown list.

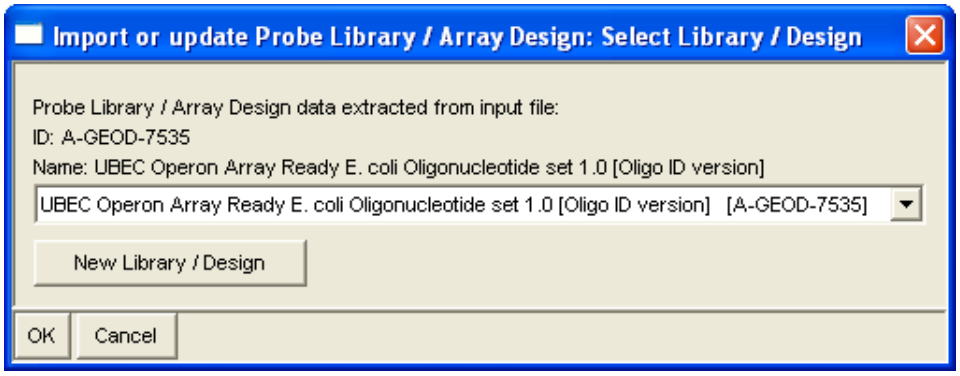


Figure 1.3: Import Probes: library selection dialog

If the respective Probe Library is not yet available in the dropdown list, it can be created now by clicking the button **[New Library / Design]**. A dialog will pop up to prompt for ID and name/description of the library (Figure 1.4). Values retrieved from input file metadata will be used as default values.

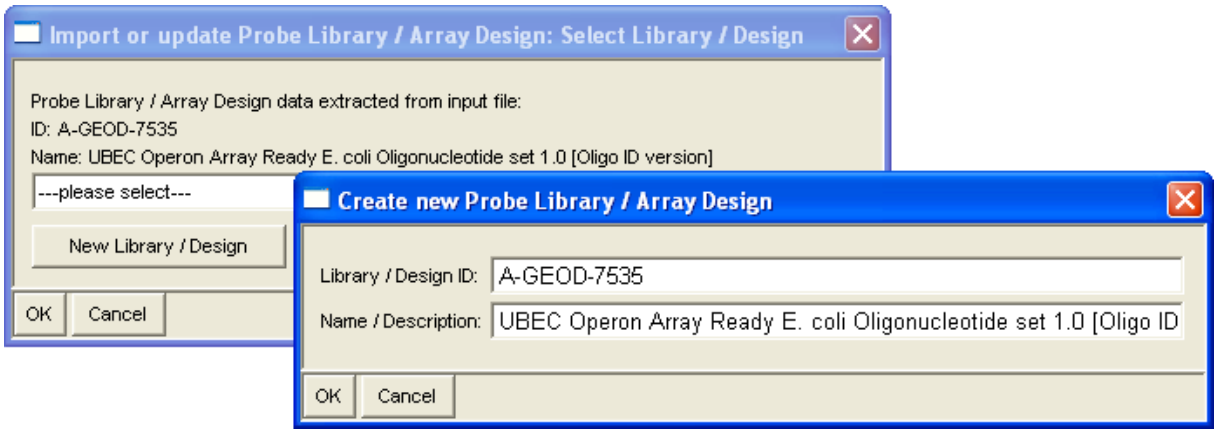


Figure 1.4: Import Probes: library creation dialog

Once you finished creating / selecting the target Probe Library, your next step is to select the columns that contain Probe IDs and sequences (Figure 1.5). Standard columns - especially those of ADF format - are preselected. After completion of this step, probes are imported into the selected Probe Library.

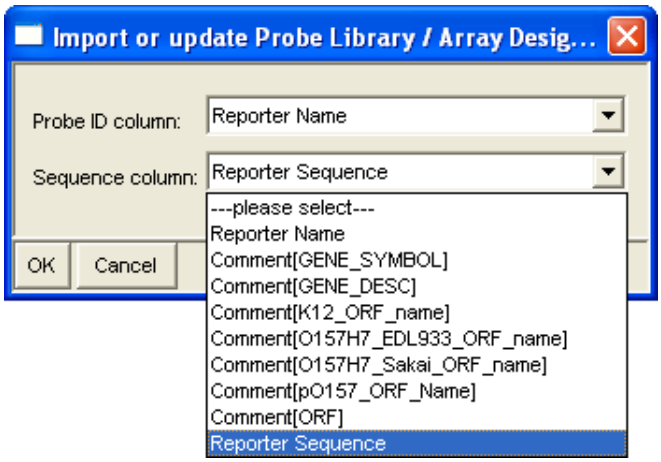


Figure 1.5: Import Probes: column selection dialog

To combine probes from multiple input files (even of differing data formats) into the same Probe Library, you can repeat the steps described above. If you do so, or whenever you like to update existing Probe Libraries to changes been made upon them, please consider the operating mode of this import/update function:

- Probes which - by their ID - are already present in the Library will be updated to the latest data found in your input file. So whenever one of your input files contains data for such a probe that differs from the data already present in the library, this probe will be updated in the library without special notice.
- Import/update does not support deleting probes. If a probe is not present in your update input file, this simply means that this probe will remain unchanged in your library. If probes are removed from an array design, it is generally recommended to create a new library to reflect this new version of array design. If you prefer

removing probes from an existing library, you have to do this manually by deleting the respective line in the `probes.txt` file in the library folder (see appendix A.1.1). (You do not need to update the .fsa file in the same directory, as this file is automatically re-created the next time you map the Probe Library to a genome.)

- Please note that after every update of a Probe Library, you have to re-map this library to your target PGDBs (see subsection 1.1.2) for this changes to take effect on future runs of gene expression data conversion.

1.1.2 Mapping of Probes to Genomes / PGDBs

Before you can align gene expression data to your PGDBs, you have to map the probes of the respective library to the genomes of these PGDBs. To do so, call

ACIB-Toolbox
└ ***Gene Expression Data Integration***
└ ***Map Probes to Genome***

A dialog will ask you to select Probe Library and target PGDB (Figure 1.6).

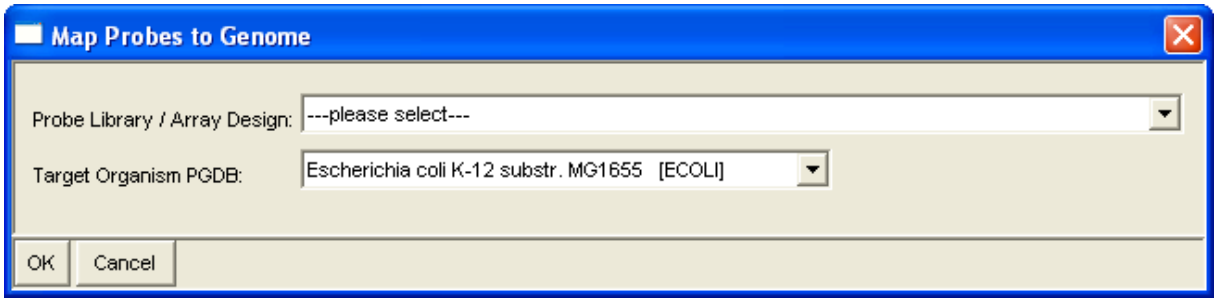


Figure 1.6: Map Probes to Genome: library and PGDB selection dialog

Probes are mapped to genomes using BLAST 2.2.25 with parameters `-F F -e 1e-3`. BLAST results are filtered to keep only hits that show $\geq 80\%$ sequence identity, which then are mapped to PGDB genes. The probe mapping results are stored in a subfolder of the PGDB's current version's data folder (see appendix A.1.2) as 3 different files:

- probe mapping: GFF format
- target gene mapping: GFF format
- mapping of probe IDs to gene IDs within the PGDB: tab-delimited text

Both GFF files can be used as external annotation tracks in the genome browser. The probe mapping GFF file displays the actual location of mapped probes, whereas the target genes GFF file displays the location of their target genes. In both GFF files, score is defined as percent sequence identity of the probe alignment, and the probe ID is used as label for the "horizontal track" in the Pathway Tools genome browser.

Note: Every time the content of a probe library changes, you also have to repeat this probe mapping step for all PGDBs the library is used for, in order to let the changes take effect on future gene expression data mappings. The same applies to changes of genome sequence or annotated genes of a PGDB, in which case all probe libraries associated to the PGDB need to be remapped. We also recommend to re-map all probe libraries associated with a PGDB after every version update of PGDBs downloaded from the registry or other external sources.

1.2 Import and Conversion of Gene Expression Data

ACIB Toolbox can import and convert gene expression data from MAGE-TAB or compatible tab-delimited text files with a column header row, a probe ID column and one or more data columns. For numerical data columns, optional replacement of NaN-values by fixed values and mathematical transformation of values are supported as well.

To import and convert gene expression data, you have to provide a file containing the definition of how to interpret and convert your data (see appendix A.1.3) along with your data files. This file can later be used as documentation of how the data was processed, or to (re)map your gene expression data to other PGDBs or updated versions of your initial target PGDBs.

To start the process of import and conversion, select
ACIB-Toolbox
└ ***Gene Expression Data Integration***
└ ***Convert Gene Expression Data***
from the menu. In the dialog that will pop up (Figure 1.7), select the Probe Library that corresponds to your experiment/data, the PGDB you want to map your data against, and the conversion definition file.

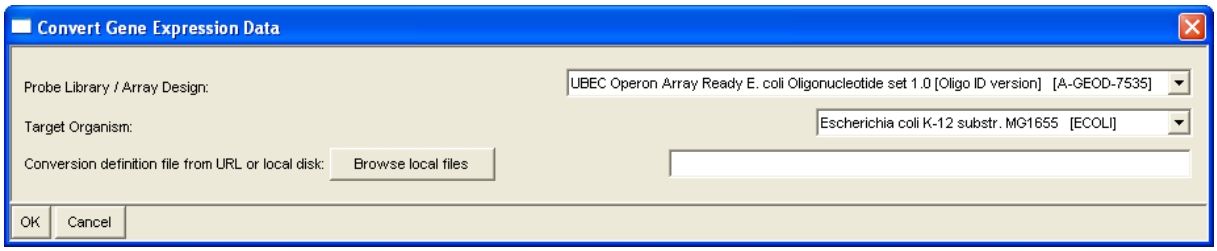


Figure 1.7: Convert Gene Expression Data: library, PGDB and input selection dialog

Gene expression data will be converted into two files:

- A GFF file containing input for displaying gene expression data tracks in the Pathway Tools genome browser.
- A tab-delimited text table file that contains in the first column the target gene IDs within the PGDB and data columns containing processed gene expression data values for each score track you defined. This file can be used as input for experimental data overlay in Pathway Tools' omics viewers.

These result files will be stored in a subfolder of the PGDB's current version's data folder (see appendix A.1.2).

Chapter 2

Annotation

2.1 Insertion Sequence (IS) Elements

IS Elements are a major cause of genomic modifications that may change a cells phenotype dramatically even during relatively short-termed experiments under non-mutagenic conditions, as recently demonstrated (Gaffé et al., 2011). To analyze IS Element abundance and activity, we integrated a systematic IS Element annotation as well as suitable cross-species comparison tools.

The IS Element taxonomy is implemented as a tree of families, groups and types within the `|Paralogous-Gene-Groups|` class of each PGDB (see Appendix A.2.1). As IS Element types behave like ordinary paralogous gene groups, they profit from all display and browsing options provided for such groups, e.g. clickable links from IS Element types to specific transposase genes and vice versa or location maps of all genes of a certain type (see also Pathway Tools User Guide). IS Element annotation is facilitated by a system of dialogs for selection of defined IS Element types as well as easy creation of new taxonomy entries. By default, the IS Element annotation of *E. coli* MG1655 is inserted into EcoCyc at program startup, so it can be browsed and used as a reference for annotating other organisms right away. (This annotation was created using IS Finder (http://www-is.biotoul.fr/) as reference database.)

When IS Element annotation is complete in all strains of interest, and orthologous genes are linked to each other across all PGDBs, a species comparison function creates tables of all IS Element loci in each organism, including information about adjacent genes; for each other organism, the orthologous IS Element locus (if present) and the percentage of sequence identity in the 5000 bp up- and downstream region of the IS Element locus are reported. A summary table lists all groups of orthologous IS Element loci and thereby allows rapid detection of shared as well as singleton loci.

2.1.1 Browsing IS Elements

Browsing IS Elements in your current PGDB can be done in various ways. The first is to open the root class of the IS Element schema by selecting

ACIB-Toolbox
└ *Annotation*
  └ *IS Elements*
    └ *Browse IS Elements*

from the menu. On the resulting page (Figure 2.1), you can click parent and child classes as well as instances to navigate back and forth between IS Element families, groups and types.

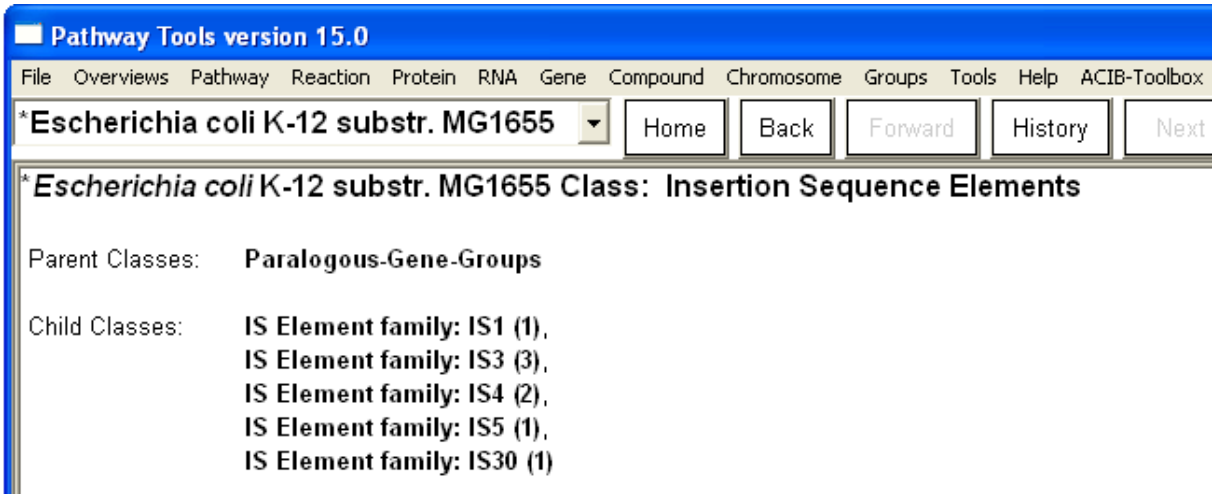


Figure 2.1: Root of the IS Element tree

IS Element type pages will display just like ordinary paralogous gene group pages (Figure 2.2). They list all genes assigned to this type and show their locations on the genome. Both the list entries and tick marks on the location map are clickable links that lead directly to the respective gene overview page. From the gene overview, you can navigate to the IS Element type page by clicking this type in the gene's "In Paralogous Gene Group" section.

2.1.2 Species comparison

Prerequistite for IS Element species comparison is that all of your PGDBs of interrest contain complete IS Element annotation as well as complete linkage of orthologous genes. To start species comparison, select

ACIB-Toolbox
└ *Species Comparison*
  └ *IS Element analysis*

from the menu. In the following dialog, select the organisms/PGDBs you want to compare. After you confirmed your selection by clicking the **[OK]** Button, you must select a target directory and enter the base name for your analysis result file set, which is referred to as {`AnalysisName`}.

The result of this analysis consists of two types of files:

- **{**`AnalysisName`**}**`-OVERVIEW.txt` - An overview over all IS Elements found in all organisms of the analysed set, grouped by orthologous genes (see Appendix A.2.2).

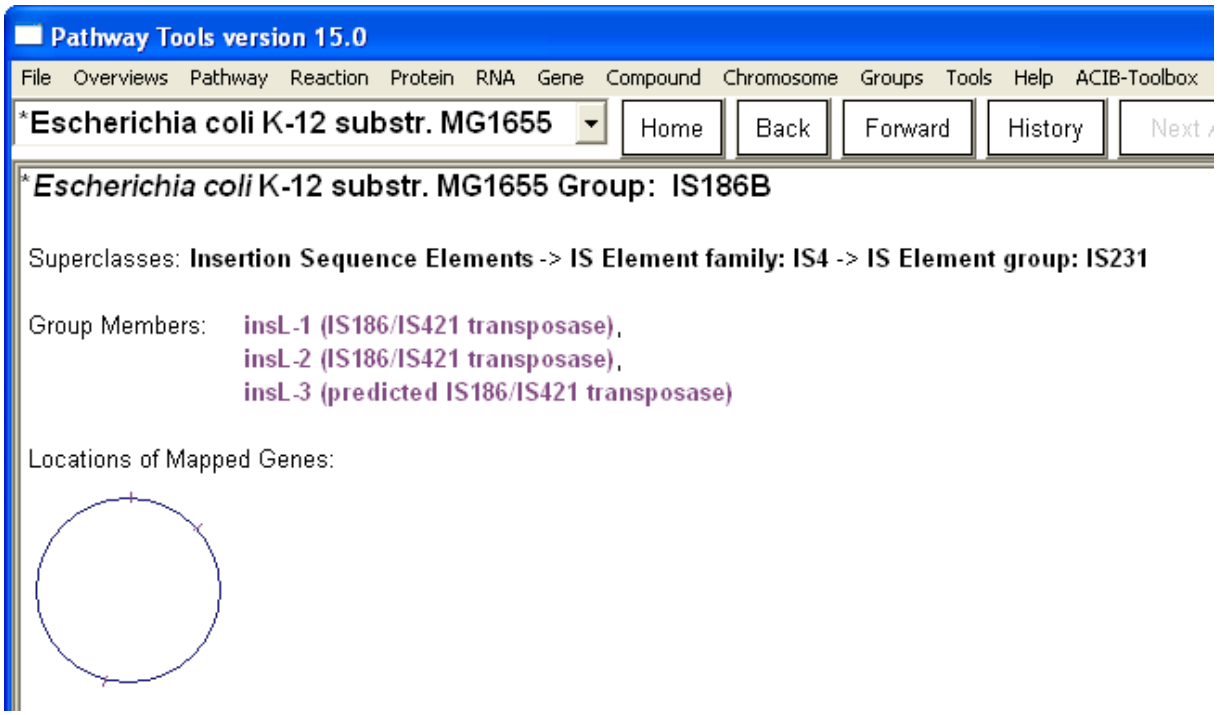


Figure 2.2: IS Element type overview page

- {AnalysisName}-{PGDB-Org*X*ID}.txt - Individual files for each organism of the analysed set, describing its IS Elements, their adjacent genes as well as their orthologs in all other organisms of the selected set (see Appendix A.2.3).

2.1.3 Exporting IS Element annotation

IS Elements of a PGDB can be exported in GFF version 2 format (http://www.sanger.ac.uk/resources/software/gff/spec.html) by calling

ACIB-Toolbox
└ ***Annotation***
  └ ***IS Elements***
    └ ***Export IS Elements to GFF***

The resulting file ISElements.gff is stored in the acib-toolbox-data subdirectory of the current PGDB's directory.

The output GFF file contains the chromosome PGDB frame ID as <seqname>, "fwd" or "rev" instead of the <source> information and "IS-ELEMENT" as <feature> qualifier. A "Note" attribute is used to store IS Element type and location, so Pathway Tools can display that information when using the GFF file as external track in its genome browser. Some sample GFF content for EcoCyc is shown below.

```
##gff-version 2
COLI-K12 fwd IS-ELEMENT 15445 16557 .  + .  Note "IS186B 15445..16557"
COLI-K12 fwd IS-ELEMENT 607288 608400 .  + .  Note "IS186B 607288..608400"
```

```
COLI-K12 fwd IS-ELEMENT 2512353 2513465 .  + .  Note "IS186B 2512353..2513465"
```

2.1.4 How to annotate IS Elements

Annotate genes as IS Elements

To annotate a gene as IS Element, you must select this gene first, so that this gene's summary page is displayed by the Pathway Tools software. Although the ACIB Toolbox usually is able to figure out the correct gene from a selected gene product (protein or RNA), it is recommended to select the gene directly by searching its name or frame ID, or by clicking it in the genome browser. This is mostly necessary is case a group of genes share the same product (sequence), in which case the selection of a gene product doesn't provide a unique gene selection.

Tip: If you have to annotate a series of genes that follow a naming scheme, you can use a little trick: Search the genes by substring by calling

Gene
└ *Search by Substring*

from the menu, select your targets in the search result and then work your way through these genes by annotating the current gene and then navigating to the next one by clicking the **[Next Answer]** button below the menu bar.

Note: In case an IS Element consists of multiple genes (e.g. for the transcriptional repressor and the functional transposase after translational frameshift), it is recommended to use the most "complete" gene of this IS Element - usually the active transposase - for annotation.

Once the gene is selected, call

ACIB-Toolbox
└ *Annotation*
└ *IS Elements*
└ *Annotate current gene as IS Element*

to open the IS Element type assigment dialog (Figure 2.3).

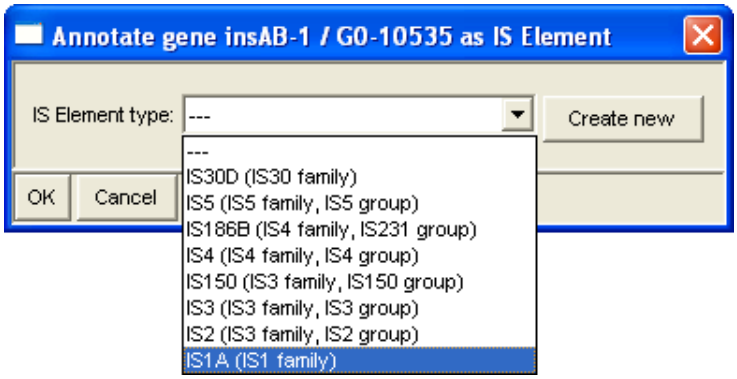


Figure 2.3: Dialog "Annotate gene XY as IS Element"

In this dialog, you can select the IS Element type from a dropdown list. If the appropriate type is not in the list, you can create it by clicking the button [**Create new**]. A dialog will pop up that allows you to select IS Element family (mandatory) and group (optional) and to enter the name of the new IS Element type (Figure 2.4). Please note that you have to enter IS Element names without whitespace; it is strongly recommended to use exactly the notation found in the IS Finder (http://www-is.biotoul.fr/).

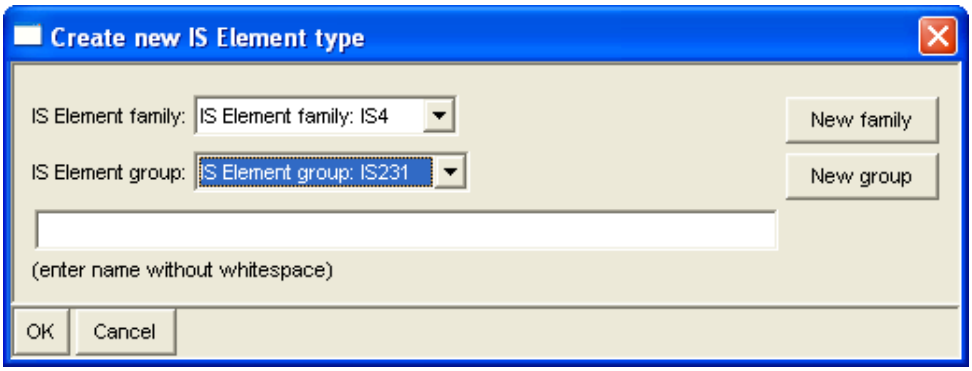


Figure 2.4: Dialog "Create new IS Element type"

To create a new IS Element family, click the button [**New family**], and enter the name of the IS Element family in the input dialog that will pop up. To create a new IS Element group, first select the family this group belongs to, then click the button [**New group**] and enter the name of the IS Element group in the input dialog.

Initialize IS Element schema for E. coli

The ACIB Toolbox offers a shortcut function to create entries for the IS Element families, groups and types most common in *Escherichia coli*. (This schema was created using IS Finder (http://www-is.biotoul.fr/) as reference database.) To use this function, select the PGDB on the Organisms Summary page or with the species selector of Pathway Tools, then call

ACIB-Toolbox
└ ***Annotation***
└ ***IS Elements***
└ ***Initialize E.coli IS Elements schema***

from the menu.

Chapter 3

Species Comparison

Pathway Tools offers a variety of species comparison features e.g. at genome, metabolism and regulation level to detect large-scale differences like missing genes or reactions. The ACIB Toolbox extends this capabilities by adding a multiple DNA sequence alignment feature that now allows detection and analysis of point mutations within genes or their upstream sequences directly from within the application.

3.1 Multiple Alignment of Orthologous Genes

An expanded CLUSTAL W alignment feature can be used to compare orthologs of a selected gene at DNA sequence level. Organisms that should be included in the alignment are selected in the same way as for display in the multi-genome browser; as an alternative, an organism selection dialog (Figure 3.1) is provided be selecting

ACIB-Toolbox
└ *Species Comparison*
└ *Select organisms to compare*

from the menu.

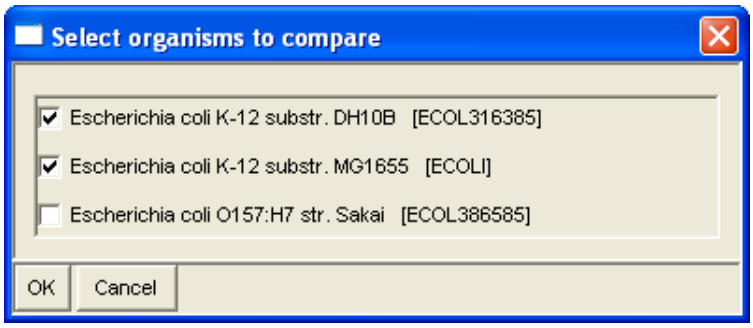


Figure 3.1: Organisms selection dialog

To start a multiple sequence alingment, first select your target gene by opening its overview page. The recommended way is to use one of Pathway Tools' gene search functions, or to select the gene by clicking it in the genome browser. Selection by gene product search is also possible, as long as the selected product is not a complex or a gene with multiple coding genes; in such a case, you have to specify your target gene by clicking the appropriate gene link on the gene product overview page. Please note that, like

for ortholog alignment in Pathway Tools' Multiple Genome Browser, ortholog links from your target gene in the current PGDB to all other PGDBs in your search space must be available (this is usually the case if using PGDBs of the BioCyc collection, with Pathway Tools' default setting to retrieve ortholog links from SRI). Once the gene is selected, call

ACIB-Toolbox
└ ***Species Comparison***
└ ***Sequence alignment of orthologous genes***

A dialog will show up to configure the alignment (Figure 3.2). In this dialog, a user-defined number of base pairs in 5' direction of the gene can be defined to be included into the alignment.

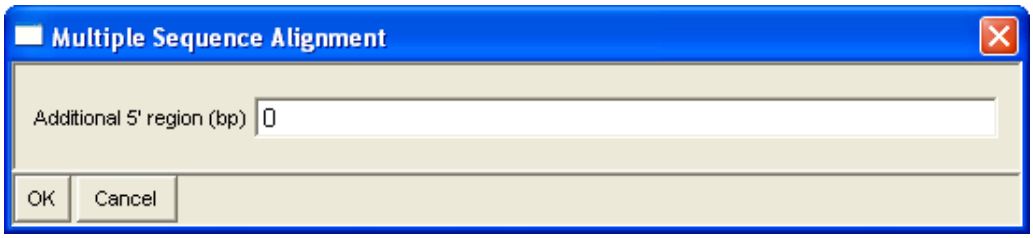


Figure 3.2: Configuration dialog for multiple sequence alignment

Multiple alignment is done by calling CLUSTAL W 2.1 using parameters `-ALIGN -OUTORDER=ALIGNED -TYPE=DNA -SEQNOS=ON`. The initial CLUSTAL alignment is processed to include a header describing all aligned genes; additional 5 sequences are highlighted by capitalization, and accurate position information relative to the start of the gene is provided. The final result is presented in a dialog window (Figure 3.3) from where it can be saved as a text file.

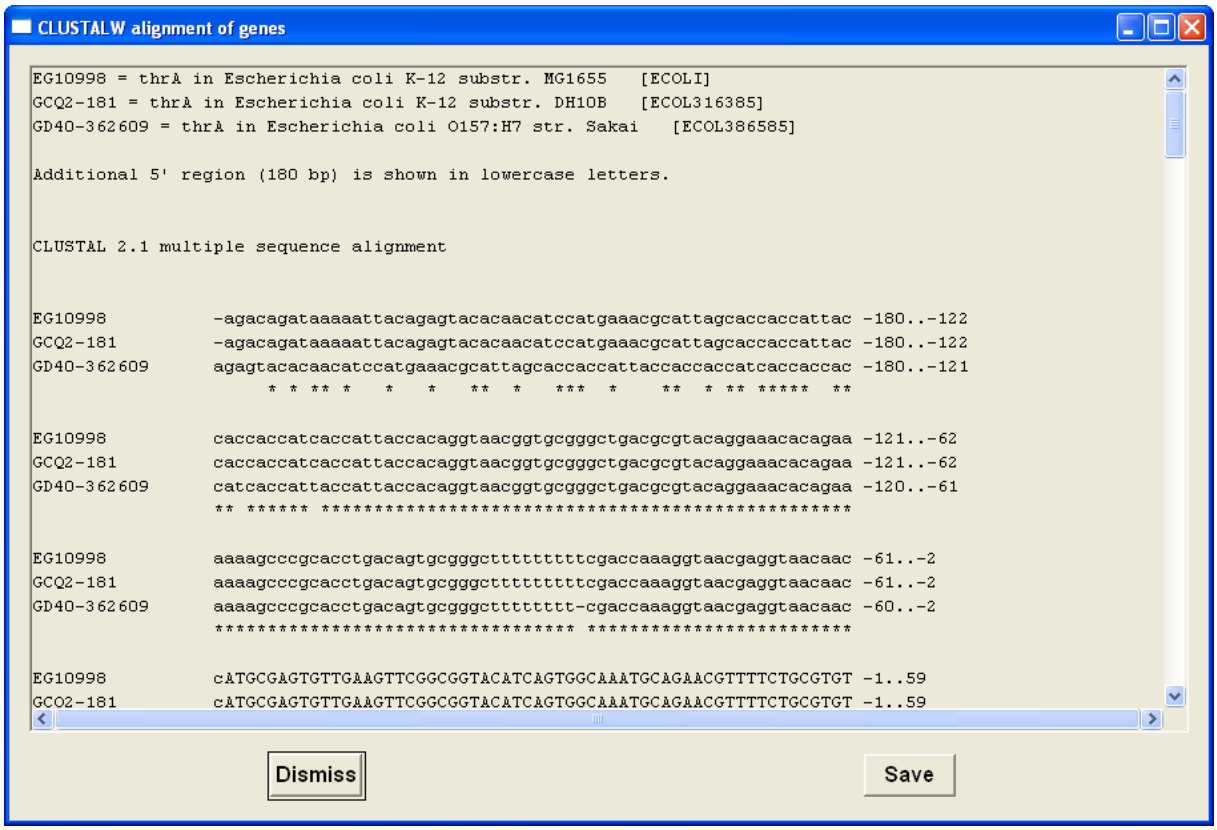


Figure 3.3: CLUSTAL alignment results

Chapter 4

Plug-in API

The ACIB Toolbox provides an API to add custom commands to the menu of Pathway Tools at application startup, or at any given timepoint during runtime. This API is also available for toolbox users who wish to integrate their own commands as menu plug-ins. All commands added using this API will be placed in the "ACIB-Toolbox" menu.

There are two important prerequisites when creating plug-in commands for Pathway Tools:

1. You must be able to programatically query and/or modify PGDBs. Therefore, you need to be familiar with the Pathway Tools API (http://brg.ai.sri.com/ptools/api/), and have some basic LISP knowledge. In case of questions concerning this part, please read the Pathway Tools documentation, contact the Pathway Tools support team (ptools-support@ai.sri.com), or attend one of the advanced tutorials SRI offers on a regular basis.

2. Once you have a self-written LISP function (that queries Pathway Tools, exports or modifies something) at hands, you must be able to gather from the user all the information your function takes as arguments. The Pathway Tools graphical user interface (GUI) relies on the Common Lisp Interface Manager (CLIM). Therefore, to create GUI components to interact with the user, you need some elementary knowledge of CLIM. There is a lot of information about CLIM on the internet, e.g.

   - a CLIM web user guide from LispWorks: http://www.lispworks.com/documentation/lw142/CLIM-U/html/climguide.htm
   - a CLIM user guide (PDF) from Franz Inc.: http://www.franz.com/support/documentation/8.2/doc/clim-ug.pdf

   We will also provide a short CLIM reference and a few examples at the end of this chapter.

4.1 Adding new Menu Commands

Adding new commands to the Pathway Tools menu is a three-step-process:

1. Implement the function you want to integrate.

2. Wrap a CLIM command around this function. This command should also provide the user interface to prompt for all arguments your function takes.

3. Use the ACIB Toolbox API to add this command to the menu.

Creating a CLIM command for your function is done using `clim:define-command`. The most simple example, where your function doesn't take any arguments (e.g. because it gets its parameters from system variables set somewhere else, or has its own, integrated user interface), looks like this:

```
(clim:define-command com-your-command-name ()
  (your-paramless-function))
```

A more detailed description of how to create a CLIM command that also provides a graphical user interface (GUI) can be found in section 4.2.

Once the CLIM command is created, it can be integrated into the menu using the plug-in API command `acib::register-plugin-menu-command`, which takes the following arguments:

- `ui-name` - Name of the command in the menu (string).
- `command` - Command name (LISP symbol).
- `&key (submenu nil)` - Submenu where to place your command:
  - To place your command directly in the "ACIB-Toolbox" menu, omit this key, or use `:submenu nil`.
  - To place your command in a simple submenu of the "ACIB-Toolbox" menu, use this key with a string (or a list containing a single string) as argument.
  - To place your command in a submenu tree, use this key with a list of strings, each of them naming a menu level.

The following example shows how to integrate your command named `your-command-name` into the menu

ACIB-Toolbox
└ ***First Level Submenu***
  └ ***Second Level Submenu***
    └ ***Menu Command Name***

```
(acib::register-plugin-menu-command
  "Menu Command Name" 'com-your-command-name
  :submenu '("First Level Submenu" "Second Level Submenu"))
```

4.2 Creating CLIM commands with GUI

To create a graphical user interface for your function using CLIM, you should be familiar with some CLIM functions and macros:

- `clim:accepting-values` - to create a dialog that prompts users to enter information
- `clim:formatting-table, clim:formatting-row, clim:formatting-cell` - to obtain a tabular layout of your labels and input fields
- `clim:accept, clim:accept-values-command-button` - to read user input
- `clim:notify-user` - to display a notification dialog

Pathway Tools itself also provides some helpful functions:

- `format-cell-text` - to display text, e.g. a label of an input field
- `accept-string` - provides a text input box
- `select-file-w-default` - to display a file selection dialog

Finally, the ACIB Toolbox offers a macro that wraps a `clim:accepting-values` dialog around the developer's UI code in the body: `acib::with-input-dialog`. Compared to `clim:accepting-values`, this macro offers layout enhancements as well as an easy tool to introduce validation of user input when the [**OK**] button is pressed. If the dialog sucessfully exists after pressing [**OK**], it will return `:ok`. The essential parameters of this macro are:

- `stream` - ...where the macro prints the dialog; your GUI components may need this.
- `&key (label "")` - The dialog title, empty by default.
- `&key (ok-button-test T)` - An expression that is evaluated whenever the [**OK**] button is pressed. Developers can use this to test if the given input is usable for the following productive function, and may include user interface components to give the user a hint how to justify his input. The expression must evaluate to `nil` if something is wrong with the user input, in which case the dialog remains open; if the user input is OK, any non-`nil` value may be returned. By default (`:ok-button-test T`) the dialog returns `:ok` whenever [**OK**] is pressed.

The code example on the next page shows how to use `acib::with-input-dialog` to create a basic CLIM dialog command that asks the user for selection of an organism/PGDB and selection or input of a file name. When pressing the [**OK**] button, some elementary parameter checking is done. If parameters are undefined, a warning message occurs, after which the user returns to the dialog. If all parameters are valid, an example function is invoked; this function is defined on top of the code example and will show the selected PGDB and file in a user notification dialog. At the end of the code example, the command is integrated into the plug-in menu. This example can also be found at:

```
{ACIB Toolbox installation directory}
└─ examples
   └─ PlugInAPI
      └─ SimpleDialogExample.lisp
```

```
(in-package :ecocyc)
(defun my-function (kb file) (clim:notify-user clim:*application-frame*
  (format nil "Using KB ~A and file ~A..." kb file) :title "ACIB PlugIn API Example"))
(clim:define-command com-example-command ()
  (let* (
      ;; Organism / PGDB: selection list and default selection
      (org-choices-alist (acib::create-org-choices-alist :include-nil-line? t))
      (org-id (org-id-of-kb (current-kb)))
      ;; File name (string), to be filled by input line or dialog
      file-name)
    (when (equal :ok (acib::with-input-dialog (stream
          :label "Example: Do something with a PGDB and a file."
          ;; When user presses "OK" button, check input parameters:
          :ok-button-test (cond
            ((not org-id)
              ;; No organism/PGDB selected: Prompt user to select an organism...
              (clim:notify-user clim:*application-frame*
                "Please select: Target Organism."
                :title "Warning" :style ':warning :exit-boxes '((:exit "OK")))
              nil) ;; ... and return nil to keep the dialog open.
            ((= (length file-name) 0) ; usually also check if file-name is valid
              (clim:notify-user clim:*application-frame*
                "Please provide: File from URL or local disk."
                :title "Warning" :style ':warning :exit-boxes '((:exit "OK")))
              nil)
            ;; All parameters valid: Return T to close dialog with status :ok.
            (T T)))
        ;;
        ;; Content of dialog: tabular layout.
        (clim:formatting-table (stream)
          ;; "Target Organism" selection row
          (clim:formatting-row (stream)
            ;; Label/prompt and spacers.
            (format-cell-text stream "Target Organism:"
              :align-y :center :align-x :left)
            (format-cell-text stream "  " :align-y :center :align-x :center)
            (format-cell-text stream "  " :align-y :center :align-x :center)
            ;; Cell containing the selection box that sets 'org-id'.
            (clim:formatting-cell (stream :align-y :center :align-x :right)
              (setq org-id
                (clim:accept `(clim:member-alist ,org-choices-alist)
                    :stream stream :prompt nil
                    :default org-id
                    :view clim:+option-pane-view+))))
          ;; Input file input/selection row.
          (clim:formatting-row (stream)
            (format-cell-text stream
              "File from URL or local disk:"
              :align-y :center :align-x :left)
            ;; Cell with button to set input file by file selection dialog.
            (clim:formatting-cell (stream :align-y :center :align-x :right)
              (clim:accept-values-command-button
                (stream :query-identifier :browse-files)
                "Browse local files"
                (setq file-name
                  (select-file-w-default (unless (url-string-p file-name) file-name)
                       :open
                       :title "Select conversion definition file"
                       :default-type "txt"))))
            (format-cell-text stream "  " :align-y :center :align-x :center)
            ;; Cell with text field for direct path or URL input.
            (clim:formatting-cell (stream :align-y :center :align-x :right)
              (setq file-name
                (string-trim *whitespace-chars*
                  (accept-string stream nil file-name
                    :width 400 :query-id :fileurl)))))))
      ;;
      ;; When dialog returns :ok, call actual worker function.
      (let ((kb (kb-of-organism org-id)))
        (with-pointer-cursor (:busy) (my-function kb file-name))))))
(acib::register-plugin-menu-command "API Example Function" 'com-example-command
  :submenu '("Examples" "Usage of PlugIn API"))
```

Appendix A

Data and Directory Structures and File Formats

A.1 Gene Expression Data Integration

A.1.1 Directory Structure: Probe Libraries and Array Designs

Probe and array design data imported by the toolbox are stored in a subtree of the global ACIB Toolbox data directory (`acib-toolbox-data`) located in the Pathway Tools user data folder (`ptools-local`). Every array design or probe library has its own folder to store meta data and probe sequences. The file and directory structure is shown below:

```
ptools-local ..... Pathway Tools user data folder
└─ acib-toolbox-data
   └─ gene-expression-data-integration
      └─ probe-libraries
         ├─ {NameOf1stLibrary} ..... directory of 1st probe library
         │  ├─ info.txt ..... meta data of this probe library
         │  ├─ probes.txt ..... probes table of this library
         │  └─ probes.fsa ..... FASTA export of probes table
         ├─ {NameOf2ndLibrary}
         │  ├─ info.txt
         │  ├─ probes.txt
         │  └─ probes.fsa
         └─ ...
```

A.1.2 Directory Structure: Mapped Probes and Experiments

Probes and experiment data mapped to a PGDB are stored in a subfolder of the PGDB's current version's data folder:

```
{PGDB_OrgID} ..... main data directory of PGDB
└─ {version} ..... current version subdirectory
   └─ acib-toolbox-data
      └─ gene-expression-data-integration
         ├─ mapped-probe-libraries
         │  ├─ {NameOf1stLibrary}.gff ..... probe mapping of 1st library (GFF)
         │  ├─ {NameOf1stLibrary}.target-genes.gff target gene mapping of 1st
         │  │  library (GFF)
         │  ├─ {NameOf1stLibrary}.target-genes.txt. target gene mapping table
         │  │  of 1st library
         │  ├─ {NameOf2ndLibrary}.gff
         │  ├─ {NameOf2ndLibrary}.target-genes.gff
         │  ├─ {NameOf2ndLibrary}.target-genes.txt
         │  └─ ...
         └─ experiments
            ├─ {NameOf1stLibrary}.... directory of experiment data for 1st library
            │  ├─ {NameOf1stExperiment}.gff. genome browser data tracks for 1st
            │  │  experiment (GFF)
            │  └─ {NameOf1stExperiment}.target-genes.txt target-gene mapped
            │     data table for 1st experiment
            └─ ...
```

A.1.3 Conversion/Track Definition File Format

ACIB Toolbox uses tab-delimited tag-value(s) text files to define how gene expression data sets are imported and converted. Each line starts with a tag that defines a certain way to interpret and/or convert input data. Arguments following a tag are separated by tabs. The following tags/functions are available:

- `EXPERIMENT-ID {experiment-id}`
  - Mandatory.
  - Defines the experiment ID used as name prefix for result files.
  - `{experiment-id}`: name of the experiment or data selection thereof; may only contain alphanumeric characters, '_' and '-'.
  - It is strongly recommended to assign a unique experiment ID to every individual conversion definition. This allows you to define and use different subsets or presentation set-ups for a single experiment. (If such different conversion definitions have the same experiment ID, they will mutually overwrite their result files.)

- `REPORTER-ID-COL {column-name}`
  - Mandatory.
  - Defines the data file column containing the reporter IDs used to link expression data and probes.
  - {`column-name`}: name of the respective column in data file.
  - Use of this tag sets the reporter ID column for the current data file and all following files. This means that `REPORTER-ID-COL` must be defined at least once within a track definition file, and that the first use of this tag must occur before switching input data files. All input data files following the use of this tag will use the same value, unless a new value is defined by using this tag again. Allthough it causes no error to define `REPORTER-ID-COL` repeated times and on arbitrary positions within a data file definition (where the last value set within said definition will actually be used), it is strongly recommended to use this tag only once at the very beginning of the input data file definition.

- `DATA-FILE {file-name}`
  - Mandatory.
  - Defines the use of a new input data file to be applied for all following `SCORE-TRACK` or `CLUSTER-SPLIT` definitions until the next use of `DATA-FILE`.
  - {`file-name`} can be absolute or relative; relative filenames are interpreted relative to the directory where the track definition file is loaded from.
  - First use of this tag must occur before the first use of
    - `SCORE-TRACK`
    - `CLUSTER-SPLIT`

- `STRAND-DISPLAY-MODE {mode}`
  - Optional.
  - Defines how to display reporter matches to the different strands of a DNA sequence.
  - {`mode`} is one of the following:
    - `both-separate-tracks` - default value, matches on both strands are evaluated and displayed on separate tracks
    - `both-common-track` - matches on both strands are evaluated and displayed together on the same track(s)
    - `forward-only` - only matches on the forward/positive strand are displayed
    - `reverse-only` - only matches on the reverse/negative strand are displayed
  - Use of this tag sets the strand display mode for all following data tracks (until the next occurrence of this tag).

- `SCORE-TRACK {data-column} {track-name} {transform-op} {NaN-value}`
  - Optional.
  - Defines a track where expression data is used as score for probes matched to the genome.
  - {`data-column`} is the name of the data file column containing expression data / scores.
  - {`track-name`} is how the track should be named in data visualization
  - {`transform-op`} is a LISP-style mathematical operation to be performed on every score value, where `~A` is the place holder for the value. Example: `(- ~A)` means to use negative score values, e.g. for the "swapped" log ratio data columns in dye swap experiments.
  - {`NaN-value`} is a number to be used to replace expression data values that can't be parsed as numbers (e.g. empty values or "n.a."); if {`NaN-value`} is empty (as is the default), such values are ignored.
- `CLUSTER-SPLIT {data-column-name} {track-prefix} {default-value}`
  - Optional, Singleton.
  - Defines a data file column to be used to split the expression data into tracks representing clusters named in this column.
  - {`data-column-name`} is the name of the column containing the cluster names.
  - {`track-prefix`} is a string prefix to the track name generated from the data column value.
  - {`default-value`} is the value to be used when no entry is found in the data column; defaults to empty, which means that such probes should be excluded from the output.
  - May occur only once within a track definition file. It is strongly recommended to use this tag before the first use of any other track definition tag.

Example:

```
EXPERIMENT-ID    My-Example-Experiment
REPORTER-ID-COL  Reporter Identifier
DATA-FILE        clusters.txt
CLUSTER-TRACK    VALUE
DATA-FILE        data-1-a.txt
SCORE-TRACK      VALUE                  first timepoint
DATA-FILE        data-1-b.txt
SCORE-TRACK      VALUE                  first timepoint (dye swap)   (- ~A)
DATA-FILE        data-2-a.txt
SCORE-TRACK      VALUE                  second timepoint
DATA-FILE        data-2-b.txt
SCORE-TRACK      VALUE                  second timepoint (dye swap)  (- ~A)
```

This will merge 5 expression data files (all located in the same directory as this definition file) that contain a probe ID column named "Reporter Identifyer" and a single data column named VALUE: clusters.txt contains cluster labels to be used to get separate tracks for each cluster; the data-*.txt files contain log ratios of two time points in a two color time series experiment with dye swap, where the negative log ratios of the "swapped" arrays are used.

A.2 IS Elements

A.2.1 IS Element annotation schema

The IS Element taxonomy of families, groups and types is implemented as a specialized subset of paralogous gene groups. This allows to use all visualization and analysis features Pathway Tools offers for such groups.

Subclasses of the `|Paralogous-Gene-Groups|` class are used to set up the IS Element schema of families and groups. IS Element types - like ordinary paralogous gene groups - are instance frames of the respective family or group classes, with the actual IS Element genes being stored in the `|Group-Members|` slot.

```
|Paralogous-Gene-Groups| ..... standard class in Pathway Tools
 |__ IS-ELEMENTS ..... root class of ACIB Toolbox IS Element schema
    |__ IS-ELEMENT-FAMILY-IS1 ..... class: IS Element family (w/o groups)
    |    |__ IS-ELEMENT-IS1A ..... instance: IS Element type
    |       |__ --> Genes ..... |Group-Members| of IS Element type
    |__ IS-ELEMENT-FAMILIY-IS3 ..... class: IS Element family (w/ groups)
       |__ IS-ELEMENT-GROUP-IS2 ..... class: IS Element group
       |    |__ IS-ELEMENT-IS2 ..... instance: IS Element type
       |       |__ --> Genes ..... |Group-Members| of IS Element type
       |__ IS-ELEMENT-GROUP-IS3
          |__ IS-ELEMENT-IS3
             |__ --> Genes
```

A.2.2 IS Element Analysis Overview File Format

The overview file is named `{AnalysisName}-OVERVIEW.txt`. Each row represents a group of orthologous IS Element genes, and the file contains the following columns:

Column Header	Description
IS-Element	IS Element type
Org*X*ID.GeneId	Frame ID of IS Element gene in compared organism ***X*** (Org*X*ID is the ID of the PGDB for this organism)
Org*X*ID.GeneName	Short name of IS Element gene in compared organism ***X***

A.2.3 Organism-specific IS Element Analysis File Format

Analysis result files of the IS Elements of an individual organism are named {`AnalysisName`}-{`PGDB-Org`*X*`ID`}`.txt`. They contain for every IS Element of the organism gene ID, name, and direction of the IS Element gene itself and the 3 adjacent genes up- and downstream of the IS Element, respectively. Additional columns show, for every other organism compared in this analysis, the percent identity of the sequence alignment of the region ±5000 bp around the IS Element (in the main organism) to the genome of this other organism, and the orthologous IS Element gene, if any.

Column Header	Description
IS-Element	IS Element type
GeneID	Gene (frame) ID of IS Element in PGDB
GeneName	Short name of the IS Element gene
LeftGene***N***.ID	Frame ID of gene ***N***=3..1 positions left of IS Element
LeftGene***N***.Name	Short name of gene ***N***=3..1 positions left of IS Element
LeftGene***N***.Dir	Direction of gene ***N***=3..1 positions left of IS Element
DIR	Direction of the IS Element
RightGene***N***.ID	Frame ID of gene ***N***=1..3 positions right of IS Element
RightGene***N***.Name	Short name of gene ***N***=1..3 positions right of IS Element
RightGene***N***.Dir	Direction of gene ***N***=1..3 positions right of IS Element
Org*X*ID.EnvScore	Percent Identity of region ±5000 bp around IS Element in compared organism ***X*** (Org*X*ID is the ID of the PGDB for this organism)
Org*X*ID.Ortholog	Frame ID of IS Element ortholog in compared organism ***X***

Direction of genes in this case means their orientation relative to the sequence/annotation direction of the respective chromosome. "Left" and "Right" mean locations with lower or higher base position number, respectively. ">" means positive (forward) orientation, and "<" means negative (reverse) orientation.

Note: Whenever, for an organism X, **Org*X*ID**.Ortholog is empty, the **Org*X*ID**.EnvScore is expected to be quite low. An EnvScore of approx. 50% or higher without an Ortholog for the same organism indicates that possibly orthologous IS Element genes are not properly linked.

Appendix B

Examples

B.1 Gene Expression Data Integration

In your ACIB toolbox installation directory, you will find data files of an exemplary microarray experiment you can use for testing the Gene Expression Data Integration features of the toolbox. Provided are experimental data files ("raw data"), an exemplary conversion definition, a color scheme that can be used for data visualization, as well as - for reference - result files obtained from probe import, probe matching and expression data conversion.

```
{ACIB Toolbox installation directory}
└── examples
    └── GeneExpressionDataIntegration ..... Example raw data
        ├── E-GEOD-6033
        │   ├── A-GEOD-4374.adf.txt ..... array design in ADF format
        │   ├── E-GEOD-6033.sdrf.txt ..... experiment data description, SDRF format
        │   └── E-GEOD-6033.processed.1 ..... experiment data folder
        │       ├── E-GEOD-6033.ConvDef.txt ..... example conversion definition file,
        │       │   user-generated, see A.1.3
        │       ├── *_sample_table.txt ..... gene expression data files
        │       └── ...
        ├── OverviewAnimationColorScheme .. color scheme used for data visualization
        └── ExampleResults ..... Results of data import and conversion
            ├── probe-libraries ..... imported array design, see A.1.1 and 1.1.1
            │   └── A-GEOD-4374
            │       ├── info.txt
            │       ├── probes.txt
            │       └── probes.fsa
            ├── mapped-probe-libraries array mapped to EcoCyc, see A.1.2 and 1.1.2
            │   ├── A-GEOD-4374.gff
            │   ├── A-GEOD-4374.target-genes.gff
            │   └── A-GEOD-4374.target-genes.txt
            └── experiments ..... converted experiment data, see A.1.2 and 1.2
                └── A-GEOD-4374
                    ├── E-GEOD-6033.gff ..... see Figure B.1
                    └── E-GEOD-6033.target-genes.txt ... see Figures B.2, B.3 and B.4
```

The example raw data - except for the conversion definition (`*.ConvDef.txt`) usually generated by the user - was downloaded from ArrayExpress. We chose the "Reference Design time-course" experiment with accession E-GEOD-6033 (Takahashi et al., 2011), as it was performed using *Escherichia coli* K-12 substr. W3110, which is closely related to *Escherichia coli* K-12 substr. MG1655, the strain of the EcoCyc PGDB built-in in every Pathway Tools release.

To use these example files, do the following:

1. Import the array design from the .ADF file as described in 1.1.1.
2. Map the newly generated probe library to the EcoCyc genome (see 1.1.2).
3. Convert the example gene expression data files for EcoCyc, using the provided conversion definition (`*.ConvDef.txt`). By now, you should have generated all the result files also provided in the `ExampleResults` directory, especially its `experiments` subfolder.
4. To visualize the expression data in the genome browser of EcoCyc, open the genome browser and subsequently load the GFF file generated in the previous step using
   Chromosome
   └ ***Add External Track to Chromosome***
   Once the tracks are loaded, you can customize your view as described in the Pathway Tools User Guide. An example view is shown in Figure B.1, created using `E-GEOD-6033.gff` found in the `experiments` subdirectory of `ExampleResults`.
5. To visualize the expression data as animation in the Genome, Cellular and Regulatory Overview diagrams, create the respective animations from the `*.target-genes.txt` file created in step 3 using
   Overviews
   └ ***Omics Viewer: Overlay Experimental Data from***
   └ ***Text File***
   In the input dialog that will pop up, select your result file as data file, as well as your preferred Overview Diagrams for the "Paint data on" option (Tip: Depending on the number of data columns/timepoints you want to visualize, it may be recommended to choose only one paint option at a time, as several large animations running in parallel might exhaust your system resources). You can use the default settings, except for:
   - Type of display: **Animation**
   - Data columns to use:
     - "**1-22**" if you want to use the full data range,
     - "**1,3,5,7,9,11,13,15,17,19,21**" or "**2,4,6,8,10,12,14,16,18,20,22**" if you want to display only one of the two replicates over the entire time range,
     - or any other selection of columns between 1 and 22 you want.
   - Choose color scheme: you can use the default scheme, define your own or load the scheme provided in the example data directory using the **[Retrieve Saved Color Scheme Parameters]** button.

Example views are shown in Figures B.2, B.3 and B.4, created using `E-GEOD-6033.target-genes.txt` found in the `experiments` subdirectory of `ExampleResults`.

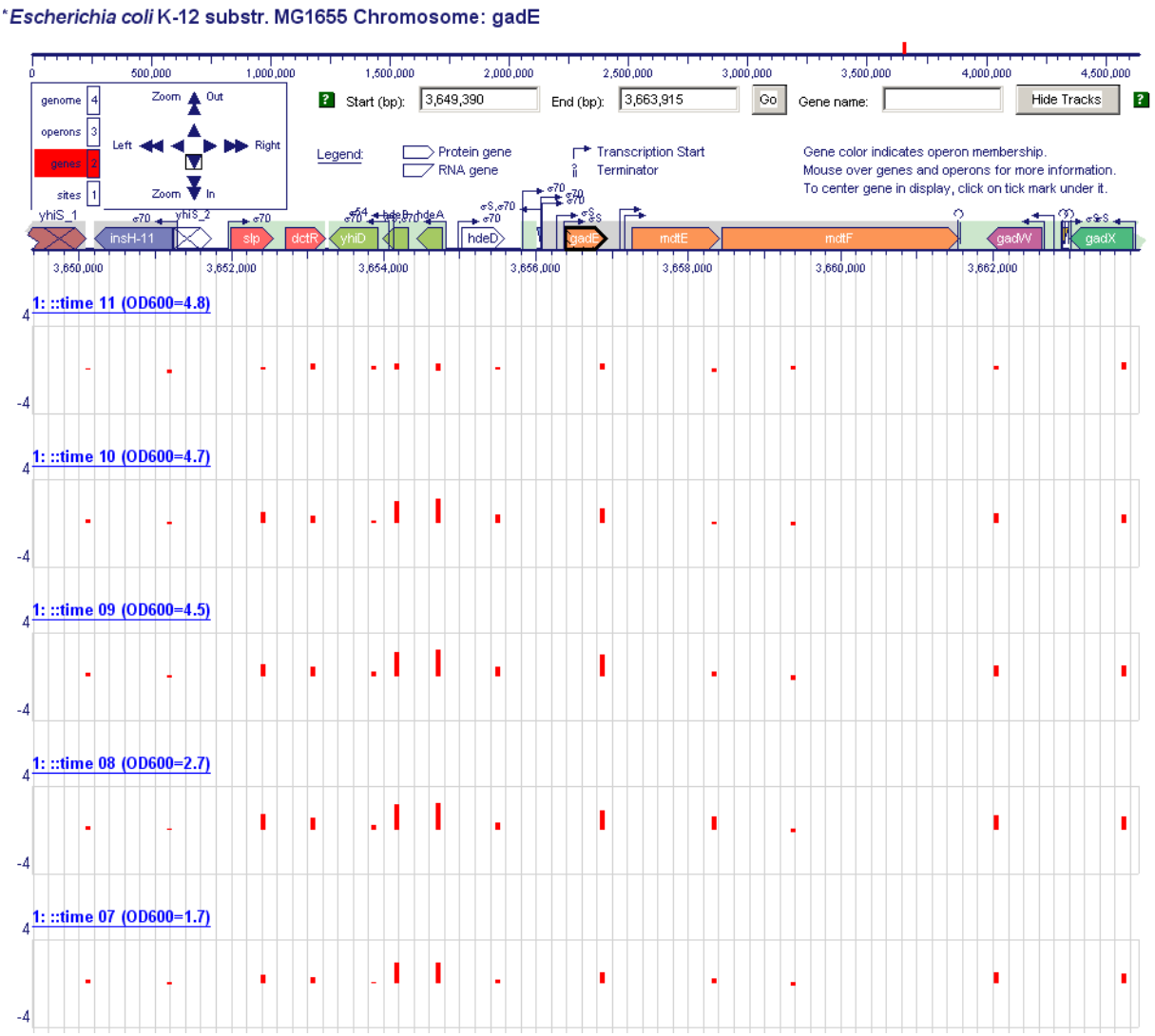


Figure B.1: Visualization of gene expression data as GFF tracks in Genome Browser. Display Options: "Show only bar graph track" with graph Y range from -4 to +4

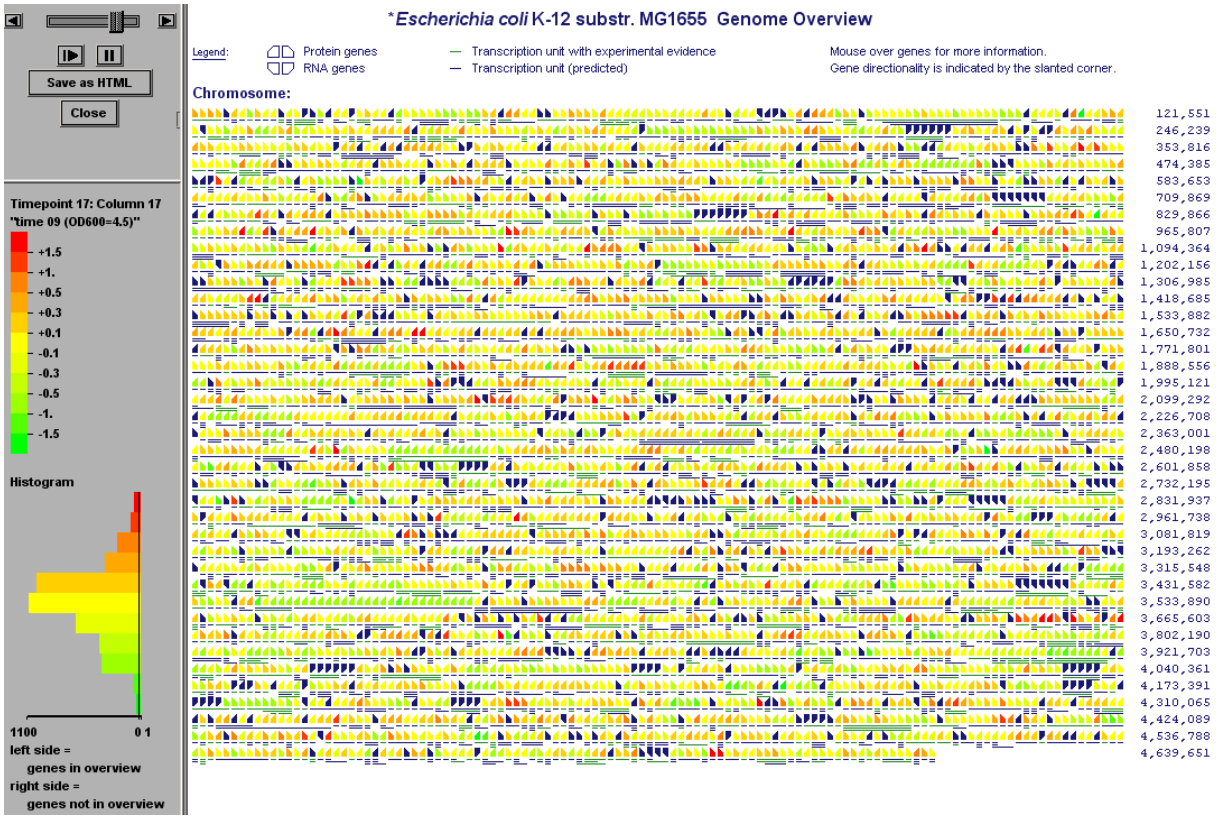


Figure B.2: Visualization of gene expression data in Genome Overview

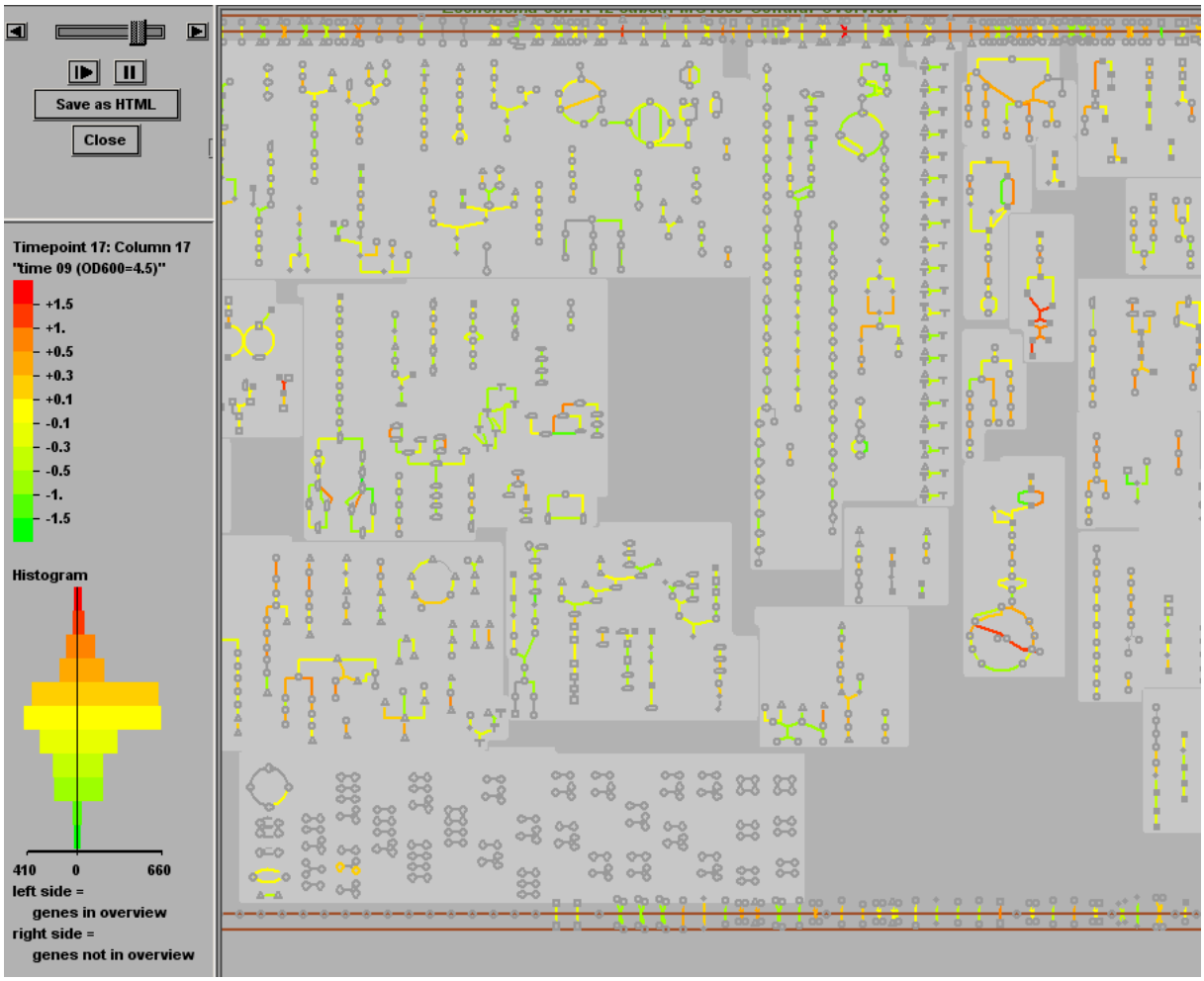


Figure B.3: Visualization of gene expression data in Cellular Overview

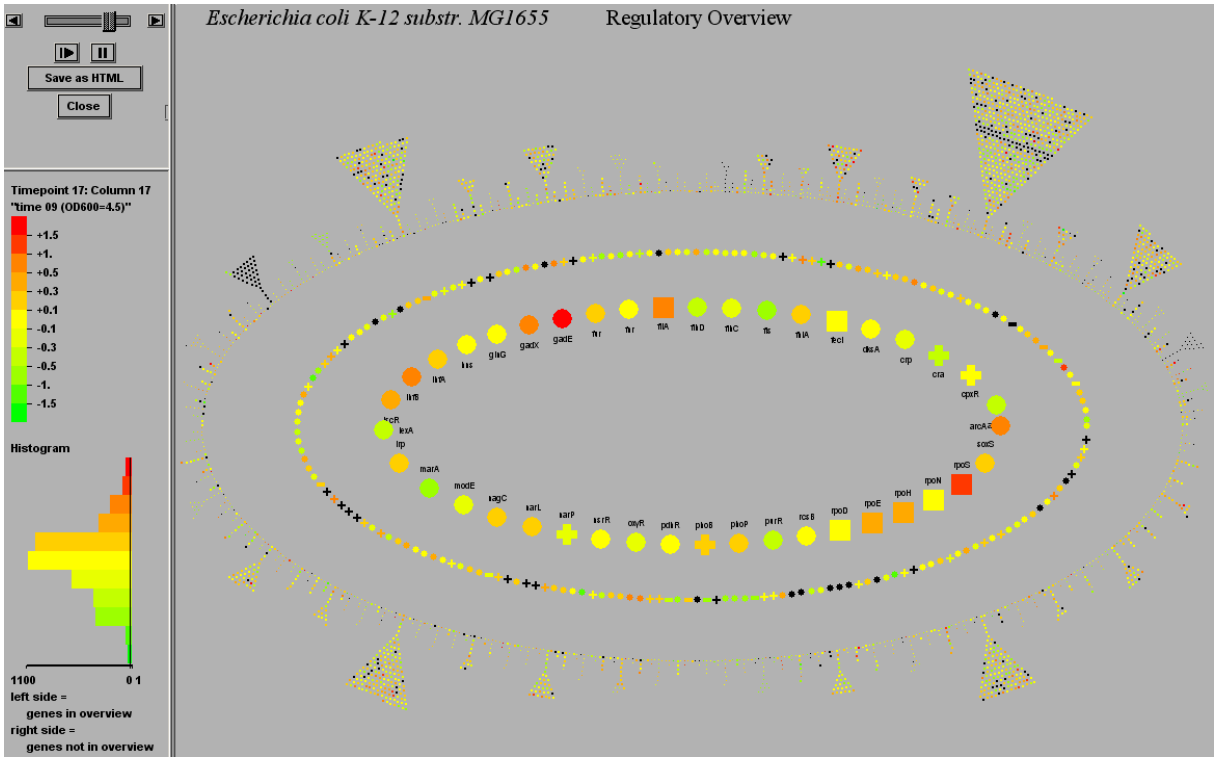


Figure B.4: Visualization of gene expression data as GFF tracks in Regulatory Overview