Jannotatix

Table of contents

1. About

1.1. Jannotatix Homepage

1.1.1. Description

Jannotatix allows you to run algorithms that create predictions or annotations on a set of sequences. Examples are repeat finders, gene finders, splicing or binding site predictors. The results are returned in a common format and can be filtered, viewed and compared using a graphical interface. The main application at the moment is motif discovery on promotor sequences. The algorithms can be either run via the web or stored on your own computer as binary programs. Jannotatix is open source, it's main data format for storing features is GFF (http://www.sanger.ac.uk/Software/formats/GFF/) .

The advantage is that you have a graphical interface for dealing with algorithms that were published in journals but had no users in mind. To make Jannotatix use a certain algorithm you simply select it from the menu or - if there is no support for it yet - write an XML-File describing the algorithm, its parameters and its output format. There is usually no programming needed to make Jannotatix work with a new algorithm, as long as the text data follows some reasonable format (e.g. something that you could also parse using awk or perl). We hope that this will increase the time-to-market for many algorithms, as they are often hidden for years on obscure webservers before people start using them.

Have a look at the tutorial to get an impression of this project.

1.1.2. Current Status

Jannotatix is still in development. There is very litle documentation, we did little testing. We are searching for people that are interesting in contributing plugins, especially motif discovery algorithm authors. For any suggestions or feedback send email to maximilian@bioinformatics dot org. [If you are searching for more information on motif discovery algorithms and a database of ~70 algorithms, please refer to Max's Masters Thesis (http://www.stud.uni-potsdam.de/~haussler/master/)].

2. Downloads

2.1. Starting Jannotatix via Webstart

2.1.1. Windows

Note:

Webstart is a handy mechanism that comes with Java. It will run programs directly from websites, will create icons on your desktop for them and will update the program whenever there is a new version. However, it is sometimes tricky to setup, but usually not in Windows. Anyways, it's your choice, you can also download the program as usual.

- **If Java is installed on your computer:** You can simply click run in Web Start and the program will be run.
- **If Java is not installed and you are using Internet Explorer:** You can do an autoinstall to automatically install both Java and Jannotatix.
- **If Java is not installed and you are using Mozilla:** I'd suggest to simply start Internet Explorer now, re-open this page and use the autoinstall for the setup, that's the simplest way as far as I know.
- **If anything of the above does not work** or you don't like to bother: Install Java from Sun's Website (http://jdl.sun.com/webapps/getjava/BrowserRedirect?locale=en&host=www.java.com) and then refer to the jarfile-download page to to do a traditional download.

2.1.2. Linux/Solaris

Linux users can choose to either install Sun's Java (http://java.sun.com/webapps/getjava/BrowserRedirect?locale=en&host=www.java.com) with Webstart included, then configure Webstart (http://lopica.sourceforge.net/faq.html#mozilla) as a helper in Mozilla and then use the usual run in Web Start link. The quicker approach, however, will be to go to the jarfile-download page and run it manually

2.2. Download Jarfile

2.2.1. How to start a jarfile

A jar-file is something like an executable file in Windows/Unix. This is all you need to run Jannotatix and, as long as you have Java installed, an easy way to run a Java program. Simply download jannotatix.jar and either double click on it (Windows/MacOS) or type `"java -jar jannotatix.jar" (Linux)`.

2.3. Jannotatix Source

2.3.1. Anonymous Checkout

If you are using the command line client, issue the command `cvs -d:pserver:anonymous@bioinformatics.org:/cvsroot login` and then

```
cvs -d:pserver:anonymous@bioinformatics.org:/cvsroot checkout
jannotatix.
```

If you are using Windows, use a client like WinCVS (http://www.wincvs.org) or TutoiseCVS (http://www.turtoisecvs.org) to check out the sources. The parameters are then: Method=pserver, Username=anonymous, host=bioinformatics.org, module=jannotatix.

2.3.2. Developer Access

Mail Max first, then have a look at the CVS Documentation of Bioinformatics.org (http://bioinformatics.org/docs/cvs/) .

3. Documentation

3.1. A short walk through the program

This text describes a sample application of jannotatix, step-by-step.

3.1.1. Start the program

Click on download on the left side and run the program, either with webstart or directly by downloading the .jar-file.

When run for the first time, it will ask for some plugins to be downloaded from the website. Select Plantcare, Transfac and AlignACE and wait until the program has finished downloading them. You can add additional plugins at any time by clicking Algorithm - Download Algorithms.

3.1.2. Download sample files and open them from your harddisk

We have put two files on this webserver: some sequences in fasta-format and some features for them in GFF-format. Right-click on the two links and save the files somewhere on your harddisk. (Webstart users simply click here to open both the program and the sample files automatically from your browser).

Click File - Open Fasta and open the first file. You can see the sequences now, aligned vertically. Click on File - Open GFF to open the sample features. They are added above every sequence, in different colors. Notice that every sequence has two lines of color-coded

features. We will call one line of features in the following a *track*.

Screenshot of the sample features

Try clicking on a feature. You will see that in the top window, on the left, properties of this feature are shown. They indicate...

1. The conkrete sub-sequence that this feature is covering
2. The *source* of this feature, i.e. the program or website that generated it
3. The *name of the sequence* this feature is located on (our sample sequences sometimes only have a simple number as a name)
4. A *score*, something the source-program attributed to this feature to indicate how well it corresponds to some model.
5. The *strand* the feature is located on
6. The *position*, given in numbers, on the sequence
7. And some optional data like the *ProfileID* (for motif discovery programs, all feature that belong to the same motif should share the same ProfileID. It they have one, a motif logo is calculated and displayed on the right side). If a program generates more than one score, they are also listed, prefixed by *ProfileScore_* and then any identifier (AlignACE, for example, that we used for the sample file, generates scores that are called MAP, so the property is named *ProfileScore_MAP*).

Whenever you click on a feature, all the other, aligned features that share the same ProfileID will be highlighted in yellow. You can use the Zoom-function from the toolbar to see all highlighted features.

If a feature was generated by a motif prediction/discovery program, then you should see a *motif logo* on the right side. A logo is a visual representation of many aligned features: The higher the letter, the less noise can is found at a position. Therefore, if a logo consists of many small letters, it is not a very good one, and its *Information Content* (IC) is very low. The maximum value for an Information Content is 2 bits at a given nucleotide position, the score that you can see above the logo is the value averaged over all positions.

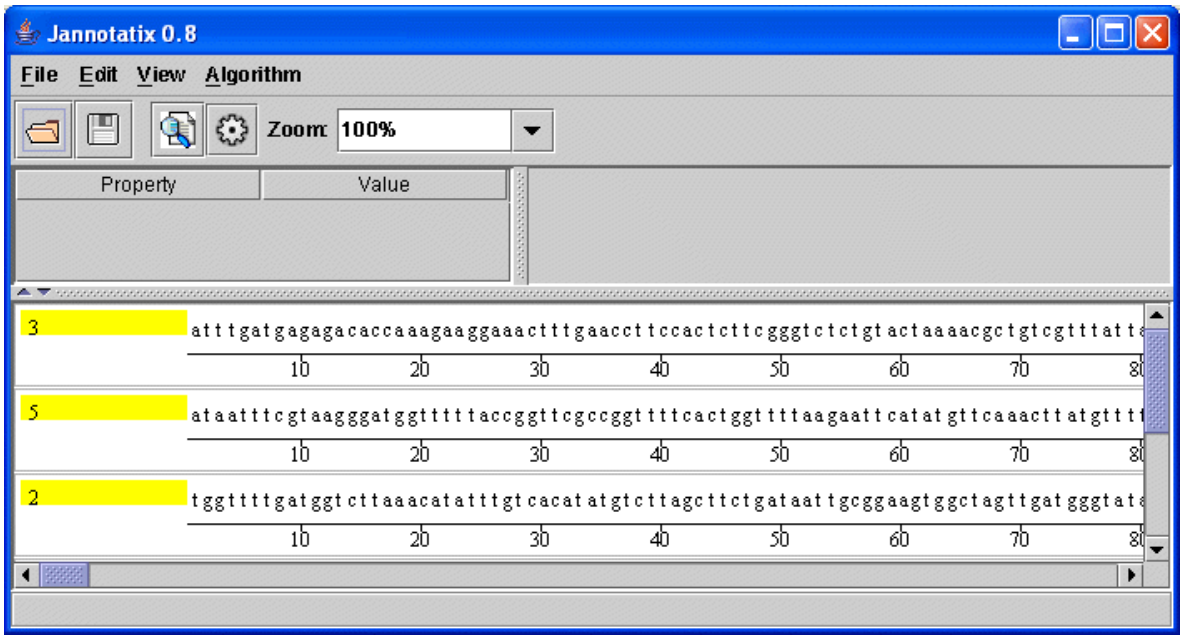
Some programs generate a wealth of additional information. PlantCare, in our example, tells you the organism, where this element was found, and some rough classification of its function (see screenshot).

3.1.3. Apply algorithms to generate new features

Click on Edit - Delete All Features to remove all features from the sample file. We will

create new ones in this step.

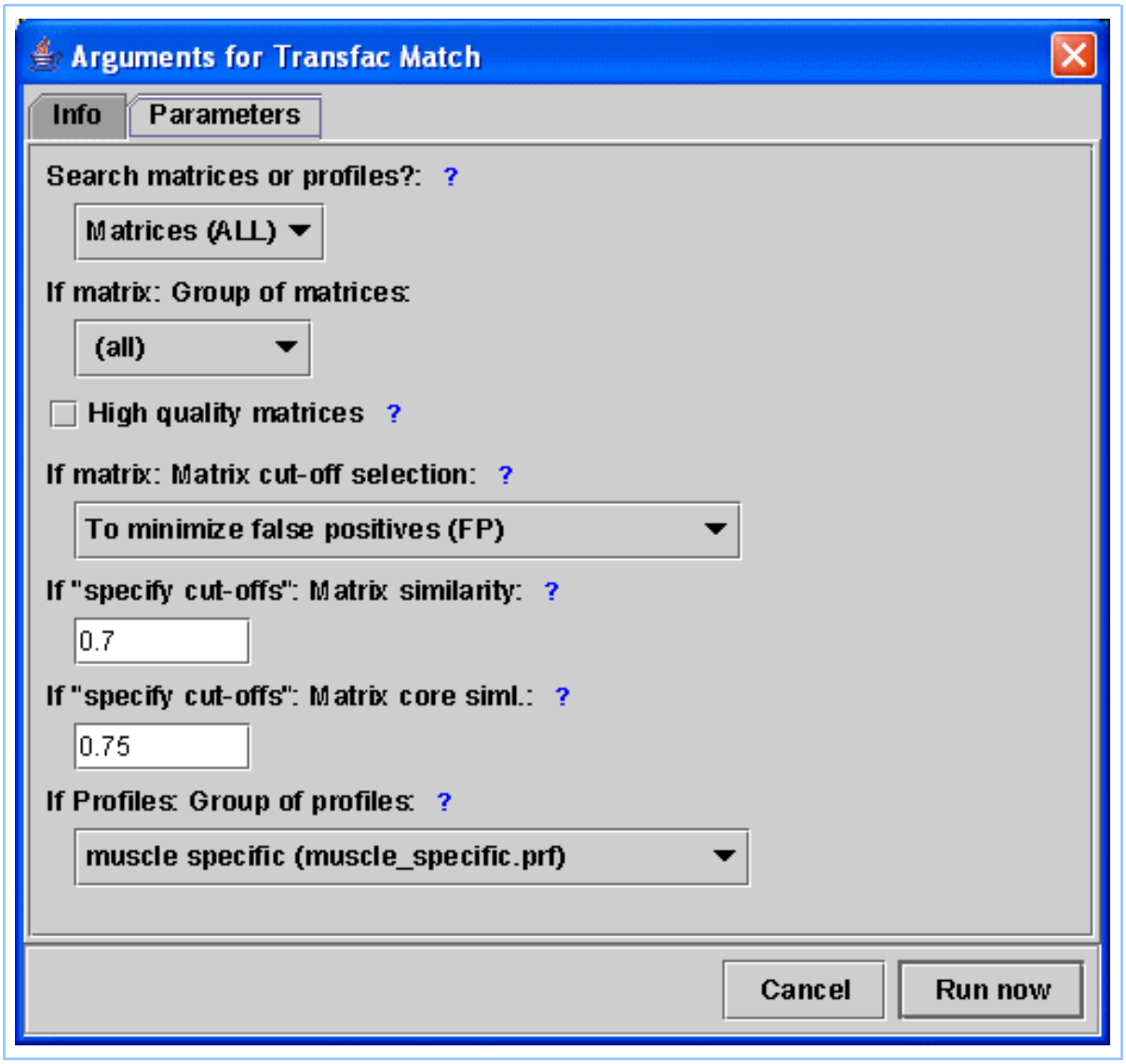
Screenshot of the empty sequences



Click on Algorithm - Transfac Match. This will run Transfac's MATCH algorithm on the sequences, just as you also could via the webform at [generegulation.com](http://www.gene-regulation.com/cgi-bin/pub/programs/match/bin/match.cgi) (http://www.gene-regulation.com/cgi-bin/pub/programs/match/bin/match.cgi) (you need a login to access, if you don't have one yet, you can get them at generegulation.com's [register page](http://www.gene-regulation.com/register) (http://www.gene-regulation.com/register)). Some details about this webpage are displayed; clicking on the underlined links that you see will open your browser with the respective address.

On the next tab, you can supply a handful of parameters, I suggest selecting High-Quality Matrices. Click on Run now. The sequences will be sent via HTTP to the webserver, which will respond with a webpage, which will be parsed and the results displayed as features on the screen. You can explore the results by clicking on them.

The Algorithm-Run Dialog



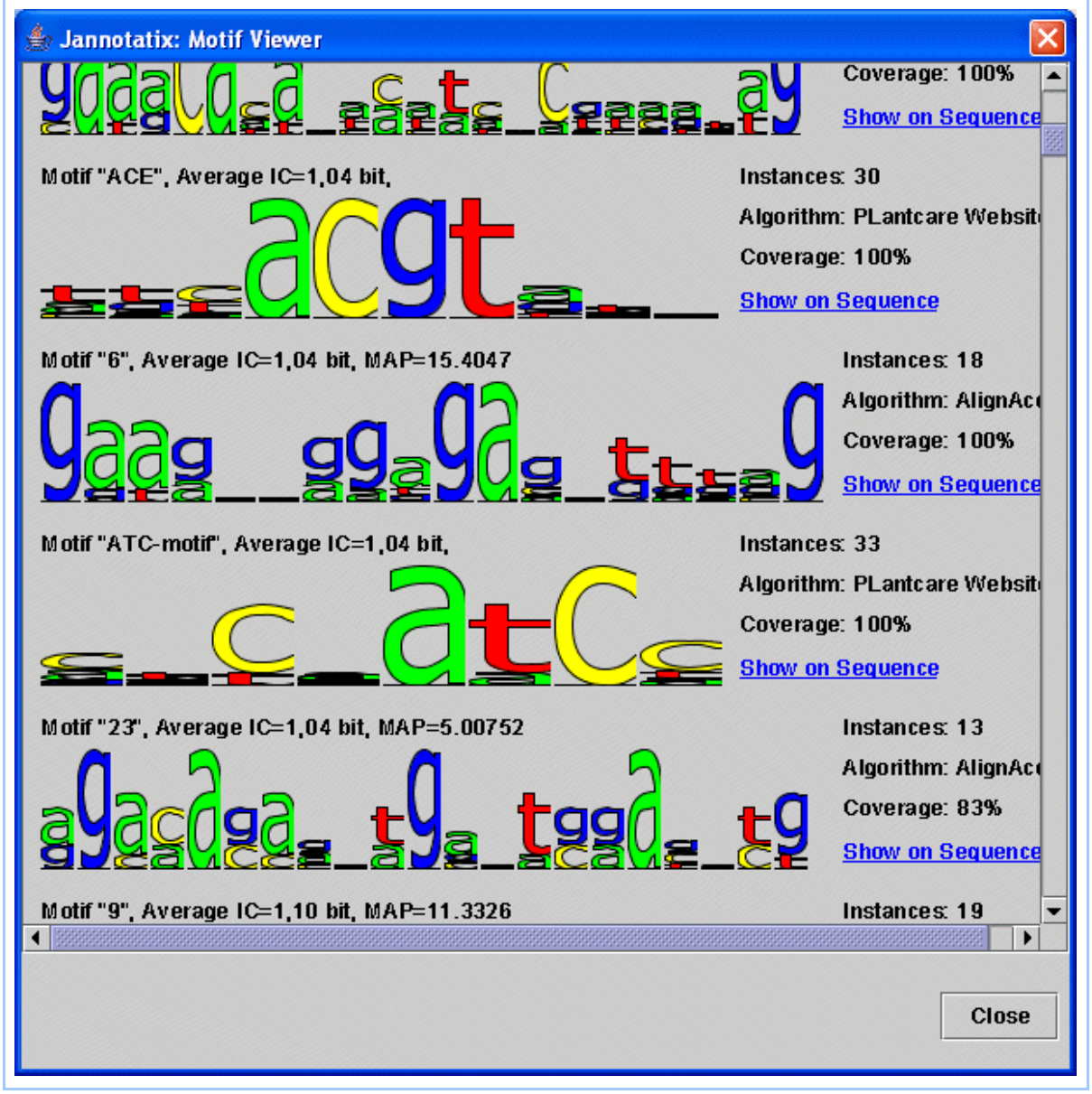
You can run additional algorithms by selecting them from the menu. Try AlignACE, for instance, which is a motif discovery program, that is run on your own computer (we only supply binaries for Windows and Linux), without any HTTP-transfers involved.

3.1.4. Analyse the results

This is something which is not at all completed. You can try *View* - *Motifs* but this is very slow (you have to wait very long until the results show up on the screen) and merely lists all

motifs across all tracks, sorted by average Information Content. We want to add here a couple features, like direct linking from the motif to the sequence view.

Screenshot of the motif viewer



Then you will need more tools: filtering very common motifs would be nice, just as comparing the tracks against some kind of reference track, to be able to benchmark an

algorithm against others (this is halfway finished in the source code). Unfortunately, Jannotatix is still in early beta stage, these functions will take some weeks or months to complete.

3.2. How to write a configuration file for Jannotatix

Title abbrev

This text describes step-by-step how to include a new algorithm into Jannotatix. We will write an .alg-file for Jannotatix. The example uses a local binary executable, pratt. We think that it is easier to follow an example than to read a formal syntax for the file format.

3.2.1. General layout of Algorithm Description Files (alg)

Note:

We will assume in the following that you already have some notion of XML-terminology. If you don't, then it might be helpful to have a quick look on any of the XML-tutorials on the web, like this (http://www.xmlfiles.com/xml/xml_syntax.asp) one.

Algorithm Description Files (alg) are simple XML-Files that consist of five parts:

- A header, which states name and category of the algorithm. It is used to build menus and when tagging features that originate from this algorithm.
- General information about the algorithm, like the author's name, the article that describes it, its website, etc. This information is just displayed to the user to give him some background about the algorithm. It is optional, this part of the XML-file can be completely left out.
- How to invoke the algorithm. Currently, this can be either a local binary executable file or a website that accepts http-post-requests and returns some result.
- A list of all parameters that the algorithm accepts, their type (number, text,...), description and help (In the future, it will include combinations of parameters that are not allowed). From this part, dialogues are built that are shown to the user who then can choose the parameters or can request additional information.
- A list of parsers that are used to convert the algorithm's results back to Jannotatix in GFF-format. Every entry of the list is an (improved) regular expression that contains named groups, a concept borrowed from Python. So you write a regular expression and mark certain parts of it to be extracted into the final GFF-file (the format of the results).

In the following, we will write a file for the PRATT-algorithm. So we go to `~/JannotatixPlugins` or to `C:\Documents and Setting\(yourname)\Jannotatix` and create a file named `pratt.alg`. Now we fire up our favorite editor and start typing:

3.2.2. The Header

This is by far the simplest and quickest part. We just write:

```
<?xml version="1.0" encoding="UTF-8"?>
<AlgorithmDescription AlgorithmName="Pratt" Category="Motif Discovery">
```

The first line is the usual XML blurb and has to be specified. The second line is the top element of our file which has two required attributes: Name and category of the algorithm.

3.2.3. General info

We give some general info about our algorithm that will be shown to the user when he chooses the algorithm on the interface

```
  <Info>
    <FullName>PRATT - Protein </FullName>
    <Description>Pratt was designed for proteins but </Description>
    <Authors></Authors>
    <HomepageUrl></HomepageUrl>
    <PubmedId></PubmedId>
    <ArticleFulltextUrl></ArticleFulltextUrl>
    <Availability>Source for Unix, Website</Availability>
    <LicenseFilename>license.txt</LicenseFilename>
    <PackageUrl
OS="Linux">http://ftp.bioinformatics.org/jannotatix/pratt.zip</PackageUrl>
    <PackageUrl
OS="Windows">http://ftp.bioinformatics.org/jannotatix/pratt.zip</PackageUrl>
  </Info>
```

Most of this information is optional and the meaning of the tags are pretty obvious, however, you're well advised to specify the PackageUrl-tag. This will be the address where you are going to store this file on the Internet later, so all users can download them from within Jannotatix's plugin manager. Create PackageUrl-directives for all operating systems that you are going to support, even if they all point to the same file in the end.

3.2.4. Invocation

Let's assume that we downloaded the PRATT program and compiled it statically on

Windows with the MinGW (http://www.mingw.org) -Compiler (which, as opposed to Cygwin (http://www.cygwin.com) allows us to compile non-open-source software as well) and compiled it on a Linux machine (good old "make"+Return is sufficient for either case). Now we just have to tell Jannotatix how to run the programs. So we write the following:

```
<LocalFileInvocation LocalBaseDir="PRATT">
  <FileName OS="Linux">pratt</FileName>
  <FileName OS="Windows">pratt.exe</FileName>
</LocalFileInvocation>
```

This will tell Jannotatix to look on which operating system it is currently running and will then run the right program.

> **Note:**
>
> If your algorithm refers to sequences only by number instead of their names from the FASTA-file you would have to add the attribute `ResolveSeqNameNumber="true"` to the `LocalFileInvocation`-tag. This will try to resolve any numbers in sequence fields to sequence names. If you don't know what this means, simply continue the tutorial, it shouldn't be very important for most algorithms.

3.2.5. The Arguments

We will now indicate all parameters that our PRATT-executable accepts. Take this paragraph from PRATT's documentation, for instance:

```
Command line:

          Pratt <format> <filename> [options]
where <format> is one of
fasta
swissprot
and <filename> is
the name of a file containing the sequences in the given format
```

So we know that the first parameter will always be "`fasta`", since Jannotatix can only export to fasta. So we write:

```
<Arguments>
<ConstantArgument Parameter="fasta">
<ShortDescription>Choose fasta as format</ShortDescription>
<Value></Value>
```

The second parameter has to be the filename of the sequences that were exported from Jannotatix. So we add to our configuration file:

```
<InfileArgument Parameter="">
    <Value></Value>
</InfileArgument>
```

There is no parameter before the InfileArgument, therefore we just leave this attribute blank. The `<Value>`-tag is needed, unfortunately, at the moment. But it is also useful, in case that we ever want to fill in real values here (see later on).

TODO: Parsers for PRATT

4. Launch Now