Documentation for the SOL database and web interface

1 Installing the SOL database and its interface

1.1 Required elements

In order to work properly the SOL interface requires these programs (followed by the version we used between brackets and by the URL where you can download them):

- Apache (2) http://www.apache.org/dyn/closer.cgi
- MySQL (4.1.2) http://dev.mysql.com/downloads/
- Perl (5.10) http://www.perl.com/download.csp
- Blast (2.2.14) http://www.ncbi.nlm.nih.gov/BLAST/download.shtml
- Melting (4.2) http://directory.fsf.org/GNU/melting.html
- MFold (3.2) http://www.bioinfo.rpi.edu/~zukerm/export/

We will not explain the installation of Apache, MySQL and Perl programs because they are often already installed on most UNIX systems. However the Blast, Melting and MFold programs deserve some explanations.

The Blast program set is used to format fasta sequence so that they can be used as an input to the SOL oligonucleotide design algorithm. Get the source code of the Blast program corresponding to your system architecture on the NCBI website. Decompress (untar) the blast-compressed file. For example with the 2.2.14 version of Blast for 32 bits Linux systems:

```
# tar -xzvf blast-2.2.14-ia32-linux.tar.gz
```

You can decompress the archive wherever you want. You need to add the path to the Blast application to your PATH variable. The executable parts of the Blast suite are located in the "bin/" sub-directory.

```
# ls blast-2.2.14/bin
```

The Perl CGI scripts used with the SOL interface need the "formatdb" executable to format the Refseq sequences properly.

The Melting and MFold programs are used to perform thermodynamic computations for the SOL algorithm. Like for Blast you will just have to decompress the archives wherever you want but if you experience some problems you may have to recompile the sources instead of using the precompiled binaries, to do that just follow the instructions gave in the archives (some compilation errors for version 2.2.13 of MFold can be solved by using an older version of gcc).Like the Blast program, you have to add the path to the Melting and MFold Binaries to your PATH variable. Furthermore, you need to export some environment variables:

```
Export MFOLDDAT to your mfold/dat directory
Export MFOLDLIB to your mfold/dat directory
Export MFOLD to your mfold directory
Export NN_PATH to your melting/NNFILES directory
```

1.2 Perl modules installation

The interface has been developed based on Perl CGI scripts and a couple of Perl modules need to be installed. If you want to know what are the modules installed on your computer, you can check your system with a Perl module called "inside", it will give you a list of what is already installed on your

computer. You can get this Perl module using the following URL:
http://www.cpan.org/modules/by-authors/id/PHOENIX

Perl modules can be easily installed from the Comprehensive Perl Archive Network (CPAN http://www.cpan.org/). In a terminal window, type one of these two command lines (as a root user):

```
# cpan
or
# perl -MCPAN -e shell
```

Once you have typed one of these two command lines, you enter the interactive CPAN mode. At first you might have to answer a few questions before installing Perl packages, but then it is very easy and simple to use because the interactive mode manages module dependencies. Under the interactive mode, CPAN waits for your instructions displaying the following prompt line:

```
cpan>
```

Type this first command to install the CGI packages:

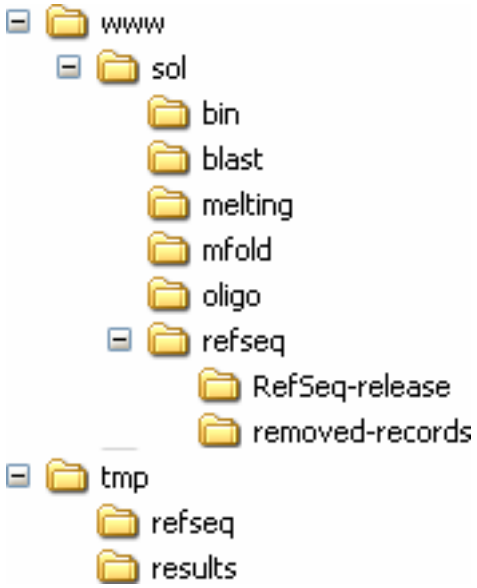
```
cpan> install CGI
```

Once CPAN runs the installation of the CGI module, you can check and install the following modules:

- About the web interface: CGI, CGI::Upload, CGI::Session
- To dump data from the database: Data::Dumper
- To build file name related to time: Time::localtime
- To send mail automatically when processes are finished: Mail::Sendmail
- To deal with Mysql DBI
- To manage file processing: File::Basename,PerlIO::gzip
- To connect to the NCBI ftp site: Net::FTP
- To read and write fasta files: Bundle::Bioperl

1.3SOL scripts installation

To enclose all the files used or created by SOL you need to create a couple of directories in the web directory of your system (this directory is often /var/www/ or /usr/lib/, from now we will note it /www/). Here are the directories we created (you can make some changes if you want but remember to report your modifications in the configuration file; see section 1.4):



Decompress (untar) the "SOL_scripts.tar" file in the "/www/sol/bin/" directory. Here are the command line you have to type under a Linux system to install all the perl scripts needed to make the SOL interface working with the Apache web server:

```
# cd /www/sol/bin
# tar -xvf SOL_scripts.tar
```

Move the uncompressed files in the bin directory.

```
# mv SOL_scripts/* ./
#rmdir SOL_scripts
```

At this stage you can check that the perl modules installation for the SOL interface has been done properly by running the Perl file mod_perl.pl in the "/www/sol/bin/" directory, if an error occurs it means that you missed the installation of a Perl module. If nothing happens and the prompt is displayed, this means that the installation worked properly.

```
# perl mod_perl.pl
```

The blast, melting and mfold directories are supposed to contain the respective programs.

The "sol/refseq/" directory will enclose all the files corresponding to RefSeq release resources and the "oligo/" directory will contain all the information concerning the oligonucleotides as the identifiers, sequences and status.

The /tmp/refseq and /tmp/results directories will contain temporary SOL result files.

Check that every Unix user will be able to access to the SOL directory and to execute the binaries.

1.4Apache and MySQL configuration

Edit the configuration file of your Apache web server, "httpd.conf"'. Usually, you will find this file in the "/etc/apache/" directory. First, set the Timeout to 5000s instead of 300s (the default value) to avoid Apache closing when the database updates.

Example of what you should find in your "httpd.conf" file (Figure 1):

```
#ResourceConfig /etc/apache/srm.conf
#AccessConfig /etc/apache/access.conf

#
# Timeout: The number of seconds before receives and sends time out.
#
Timeout 5000

#
# KeepAlive: Whether or not to allow persistent connections (more
# one request per connection). Set to "Off" to deactivate.
#
KeepAlive On
```

Figure 1: Content of the "httpd.conf" Apache web server configuration file displaying the line about the web server timeout. This means the number of seconds the Apache web server will wait for Perl script answers. If after 5000 seconds not any command has been send by the script to the web server the connexion is closed.

After each modification made in the "httpd.conf" file, save it and then restart the Apache web server so that it takes the modification into account:

```
# apachectl restart
```

Ensure the web server user (www-data, httpd or apache...) is the owner of all the SOL directories. If it is not the case, change the directory permission in order that the Apache application can read and write in the SOL directories. Example if the Apache server owner is called "apache":

```
# chown -R apache:apache /www/sol/
# chown -R apache:apache /tmp/results/
# chown -R apache:apache /tmp/refseq/
```

Now that Apache is configured, you need to create and configure a MySQL account for the SOL program. We created a user named “SOL” identified by the password “sol_sql” and granted him all privileges on a database called SOL from localhost (we advise you to read the dedicated MySQL documentation section : http://dev.mysql.com/doc/refman/5.0/en/)

You can use the followings commands:

```
Mysql > GRANT ALL PRIVILEGES ON *.* TO SOL;
Mysql > SET PASSWORD FOR SOL = PASSWORD(‘sol_sql’);
Mysql > create database SOL;
```

Then type the following command line in your terminal window to create the database structure (you do not need to be logged as a root user):

```
$ mysql -u SOL –psol_sql SOL < /www/sol/bin/sol_db_structure.sql
```

If no error message is displayed, the SOL database is created.

Furthermore, you can save your database with the command

```
$ mysqldump -u SOL –psol_sql SOL > savedfile.sql
```

And load a saved database with the command

```
$ mysql -u SOL –psol_sql SOL < savedfile.sql
```

1.5SOL configuration

The "webini.pm" file contains a lot of parameters for the customisation of the SOL interface like the database connection parameters, the NCBI website connection parameters, and information concerning the web server installation. You need to change the various paths mentioned in the "webini.pm" file if they do not suit your system (don't forget to also modify the ENV.sh file). Here is an example of what you can find in the "webini.pm" file (Figure 2):

```
#Server
$ENV{'REQUEST_METHOD'} = 'POST'; #environment variable for the server
$serveur=$req->server_name; #picks automatically your server name
$racine_abs="$serveur/sol"; #web path to your SOL folder
$racine="/sol";
$doc_root="/var/www/sol"; #absolute path to your SOL folder
$ENV{'DOCUMENT_ROOT'} = $doc_root; #environment variable for the server
$ENV{'PATH'} = '/bin:/usr/bin:/sbin:/usr/sbin:/usr/local/bin';
$smtp="biologie.ens.fr"; #email parameters
$mel='slemoine@biologie.ens.fr';#email adress

#directory
$tmp="/tmp"; #temporary folder (has to be big enough to avoid saturation)
$perl_data="/tmp";#temporary folder to save the structured informations about
$tmp_refseq="$tmp/refseq";#Where total RefSeq sequences are saved before the
$oligo_dir="/var/www/sol/oligo";#Oligo file directory
$refseq_dir="/var/www/sol/refseq";#RefSeq sequence (catalog, sequence...)

#formatdb
$formatdb='/var/www/sol/blast/bin/formatdb'; #formatdb tool path

#Sol parameters
$sol_path="/var/www/sol/bin/MASSOL.pl";
$env_path="$doc_root/ENV.sh";
$result_path="$tmp/results/";#Sol result files before insertion in the database
$nspe_file="$tmp/results/unspecific_oligo_ids";
```

Figure 2: Example of what you can find in the "webini.pm" configuration file. It gathers various parameters that must be set up to fit your system installation

1.6First steps

When this basic configuration is done you will have to insert manually the RefSeq database corresponding to the RefSeq version for which your oligonucleotides have been designed (see section 3.2.1). Then, if this version is not the latest, you will have to update manually your RefSeq database using all the RefSeq versions from the one you have designed your oligonucleotides with to the latest one (see section 3.2.1).

After each update you will be given a results summary with the number of inserted, invariant, updated or suppressed sequences. To check your own results here are the figures you should have.

Status\Version	10 to 11	11 to 12	12 to 13	13 to 14	14 to 15	15 to 16	16 to 17	17 to 18	18 to 19
New	228	3034	262	215	33590	850	17252	72	318
Invariant	25999	21052	26433	26443	18581	54276	20480	45092	43110
Updated	134	2869	261	245	4740	520	9317	152	951
Suppressed	158	2440	261	268	3582	2115	25849	1805	1255

Then you will have to insert the oligonucleotides designed by the SOL algorithm. To do so, compress the files created by the SOL algorithm (.res files) to a tar.gz archive and submit it to the “Database insertion of designed oligos” form in the “SOL oligos design” page (see section 3.3.1).

Finally you will have to check oligonucleotides specificity using every subset of new RefSeq sequences added between each update, starting from the one you have designed your oligonucleotides with to the latest one (see section 3.2.4).

2 The SOL database

2.1SOL database diagram

The SOL database diagram is displayed on Figure 3 and graphically described the 4 tables found in the SOL database created using the "SOLskeleton.sql" file (see section 1.4 for details). The definition of each field is given below.

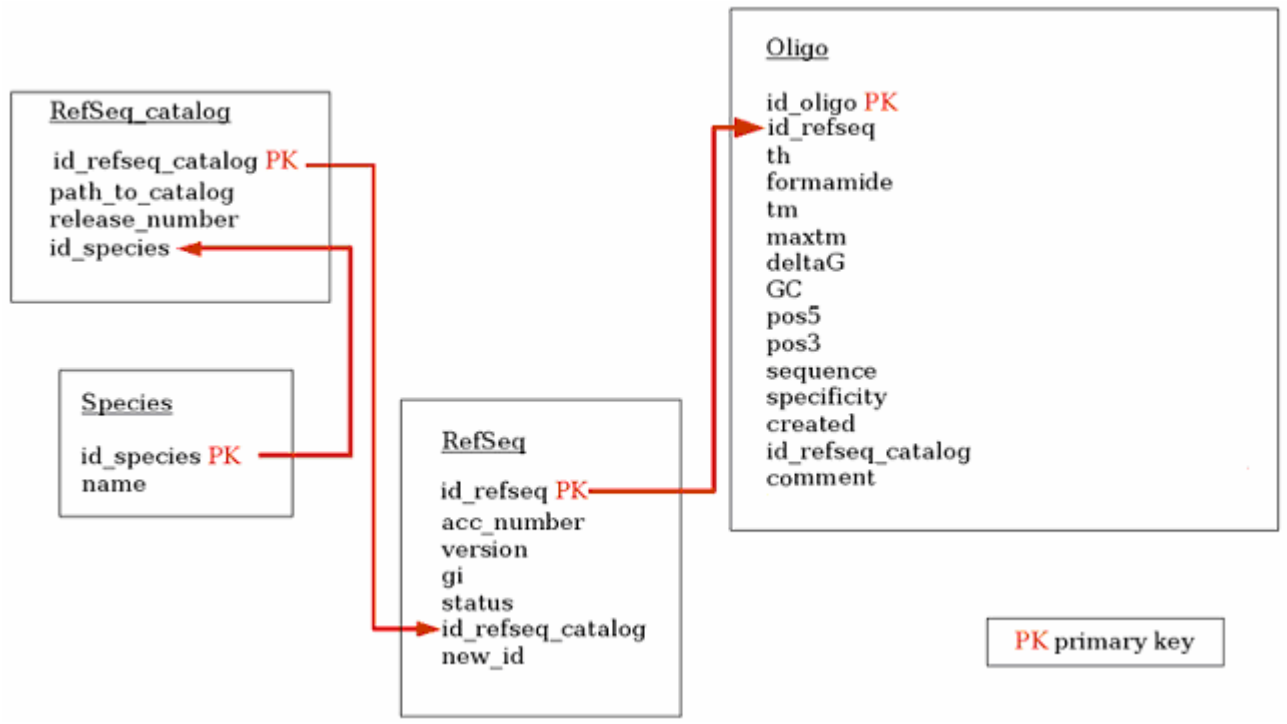


Figure 3: The SOL database diagram showing tables and fields as the relation/link found between fields. The symbol PK underlines field working as primary key for the table and U for field with unique identifier.

2.2SOL table description

Here is a detailed explanation the different fields contained in each table of the SOL database.

The "Species" table

- id_species: numerical identifier of the organism in the database
- name: the name of the organism (for example: Mus musculus, Homo sapiens)

The "RefSeq_catalog" table

- id_refseq_catalog: the numerical identifier of the Refseq version in the database
- path_to_catalog: the path to access where the "Refseq_release.catalog" file is saved locally on the web server
- release_number: The Refseq version number
- id_species: database organism identifier (link to the table Species)

The "RefSeq" table

- id_refseq: numerical identifier of the sequence in the database
- acc_number: RefSeq accession number
- version: version of the RefSeq accession number
- gi: Refseq gene identifier
- status: sequence status (NEW, UPDATE, UPDATED, REPLACED, PERMANENTLY SUPPRESSED, TEMPORARILY SUPPRESSED, INVARIANT)

- id_refseq_catalog: numerical identifier that describe the database RefSeq version (link to the table Refseq_catalog)
- new_id: new sequence numerical identifier, link to the sequence entry that replaces it or 0 if the sequence is up to date

The "Oligo" table

- id_oligo: numerical identifier for each designed oligonucleotide in the database
- id_refseq: numerical identifier of the oligonucleotide's sequence in the database (link to the table Refseq)
- th: the olgigonucleotide hybridisation temperature
- formamide: formamide percentage used as a design parameter by the SOL algorithm (usually equal to 0)
- tm: the melting temperature for the oligonucleotide
- maxtm: melting temperature of the best non-specific oligonucleotide
- deltaG: the Delta G value of the sequence's oligonucleotide duplex
- GC: the GC percentage of the oligonucleotide
- pos5: coordinate of the oligonucleotide on the 5' side of the coding sequence
- pos3: coordinate of the oligonucleotide on the 3' side of the coding sequence
- sequence: the oligonucleotide sequence designed by SOL
- specificity: oligonucleotide specificity (1 if specific, 0 if not). Check the section of the documentation for details about the specificity verification.
- created: the date when the oligonucleotide was added to the database
- id_refseq_catalog: the Refseq version identifier on which the oligonucleotides has been designed in the database (link to the table Refseq_catalog)
- Comment: contains the informations on the gene specific for the probe

3 The SOL interface

3.1 Overview of the web SOL interface

When connecting to your web server to access the SOL interface, normally located at the following address: http://your_server_address/sol/bin/index.pl, you will display the index page as in Figure 4.

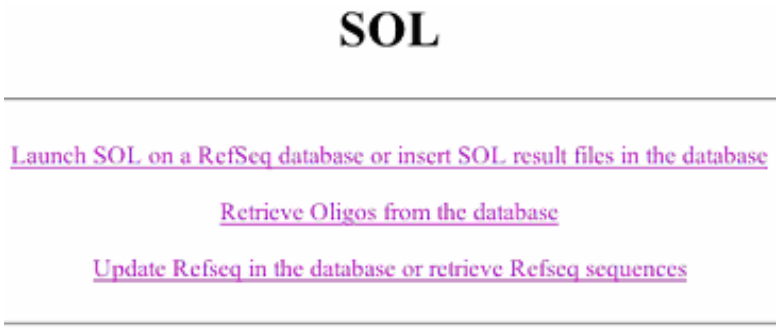


Figure 4: The SOL interface web index page that displays the choice between the three various access modes available.

From the interface you can either launch the SOL oligonucleotide design algorithm using the web interface and insert the obtained result files into the database (see section 3.3 for details), you can also query the database to retrieve designed oligonucleotides using accession number and then follow the evolution of the reference (RefSeq) database (see section 3.4 for details), and finally it is also possible to manage RefSeq updates using the web interface (see section 3.2 for details). A diagram of all the Perl scripts involved in the SOL web interface is described on Figure 5.

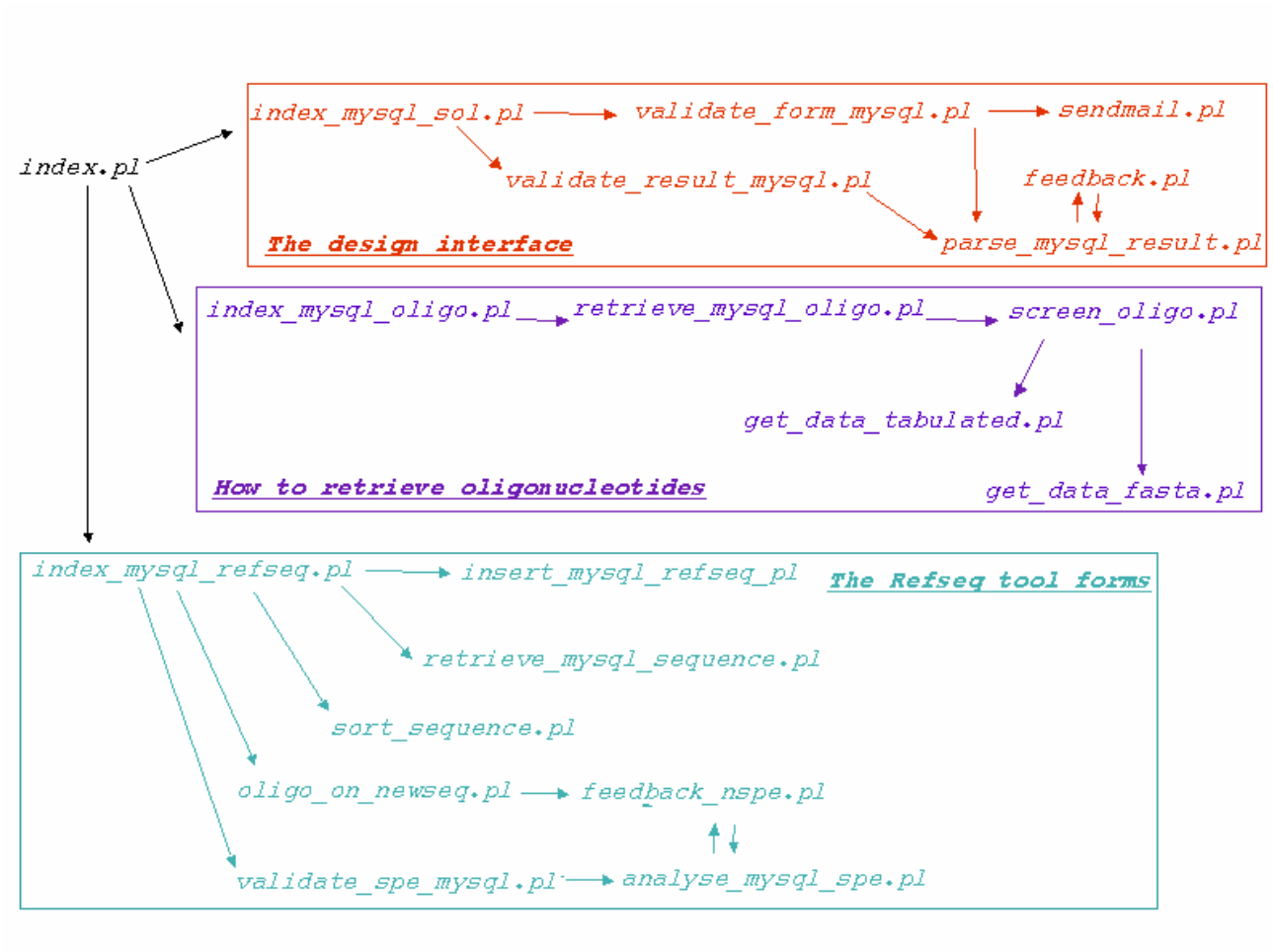


Figure 5: The SOL Perl script organization. This diagram display all the Perl scripts used in the SOL web interface with the relations involved between all Perl scripts. Arrows indicate the dependencies and the way each Perl script is launched.

3.2The Refseq tool forms: "Refseq utilities for SOL"

3.2.1 Update the database SOL database: "RefSeq database Update"

The Perl script "insert_mysql_refseq.pl" aims at updating the status of the sequences referenced in the database (the field status of the table "RefSeq" see section) at each new release of RefSeq version. The sequence types take into account in the SOL database are the XM_ and the NM_ sequences, that is to say, the predicted and validated transcripts respectively. RefSeq versions are published on the NCBI website at the following address: http://www.ncbi.nlm.nih.gov/RefSeq/, a release announcement mailing list is available. The database update is based on two files that can be retrieved from the Refseq FTP site when a new version is available (ftp://ftp.ncbi.nih.gov/refseq/release/). These files are:

- "RefSeq-release.catalog". This file references all the sequences that are part of the new RefSeq release, without separating species or type of sequences. The sequences referenced can be protein, cDNA, and genomic sequences, predicted or validated.
- "release.removed-records". This file located at ftp://ftp.ncbi.nih.gov/refseq/release/release-catalog/ traces the sequences that have been suppressed (permanently or temporarily) or replaced. In the case of replaced sequences, it gives the reference of the new sequence that replaces the old one. This file, has the first one, also contains all the organisms of the RefSeq project and mixes the sequences types.

The web interface allows you to choose the way you want to retrieve the RefSeq files. They can be retrieved directly from the FTP site (if you are up to date with your version history of the RefSeq release) or loaded manually (if the version of RefSeq you want to load in the database is not the current one, for example the first time you run the update). You also have to set the organism you want to get the sequence for. Currently, two organisms can be chosen, Mus musculus and Homo sapiens, but other organisms can be easily added (see section 4).

Once these options are chosen, the script can be launched. To update the database, the script "insert_mysql_refseq.pl" will parse the "RefSeq-release.catalog" and "removed-records" files and filters the XM_ and NM_ sequence identifiers corresponding to the selected organism. These identifiers will be compared to the previous database update identifiers. The RefSeq identifiers are then sorted according to 4 classes in 4 different files as followed: the new sequences, the updated one, the suppressed sequences and the unchanged sequences. These files will be used later to build fasta files for SOL oligonucleotide design (see section 3.3).

3.2.2 Retrieve the sequences in a fasta file: "RefSeq sequence Update"

During the database update, only RefSeq identifier files were retrieved. In order to work properly, the SOL oligonucleotide design algorithm needs fasta files containing the reference sequences. The online form retrieves the sequences for the selected organism and builds the needed fasta file using the script "retrieve_mysql_sequence.pl".

The sequences of one specific organism cannot be retrieved from the NCBI FTP site. Organisms are grouped together and sequences are merged in the RefSeq identifier files. For example, Mus musculus and Homo sapiens are part of the vertebrate-mammalian organism group and their sequences are merged in the "vertebrate-mammalianXX.tar.gz" files (ftp://ftp.ncbi.nih.gov/refseq/release/vertebrate_mammalian/). When the script "retrieve_mysql_sequence.pl" is loaded, the "vertebrate-mammalianXX.tar.gz" files are retrieved from the RefSeq FTP site and saved in a temporary folder on your computer (this folder has been set in the "webini.pm" file see section 1.5). The compressed files are parsed without being decompressed so that the sequences you are interested in can be picked and copied in a fasta file. The RefSeq release version is used to name this output fasta file, for example, "mouse13_rna.fna" for the 13th RefSeq release.

This file is next formatted with the "formatdb" Blast command so that the SOL algorithm can use it directly as its reference sequence database. This file contains all the sequences referenced in the new RefSeq version.

3.2.3 Build a sequence subsets in fasta format: "Build a RefSeq subset file"

During the database update process, subset files with RefSeq identifiers were generated for the new, suppressed, updated and unchanged sequences (see section 3.2.1). If you want to design oligonucleotides for the new sequences appeared in the 13th Refseq release for example, you will need the new identifier subset file called "new_RefSeq-release12.catalog_RefSeq-release13.catalog_Mmus" (new sequences appeared between the version 12 and 13 of RefSeq) and the complete sequence fasta file, "mouse13_rna.fna" from the 13th RefSeq release for the oligonucleotide design. Once the script "sort_sequence.pl" is launched using the online interface, the sequences will be picked according to their identifier in the identifier subset file, and then copied in a fasta file called "SB_new_RefSeq-release12_RefSeq-release13" and next formatted using the "formatdb" command to be used directly in the SOL algorithm. This "SB..." file is saved in the "sol/refseq/" folder created on the web server (see section 1.3)

.

3.2.4 Check the oligonucleotide design specificity

As oligonucleotides are designed according to one specific release of the RefSeq reference resource, they can become unspecific when a new RefSeq version has been made available. This can be explained by two reasons:

- If some sequences are suppressed, their oligonucleotides have no reasons to be referenced anymore. So these oligonucleotides must not be retrieved anymore from the database, even if the suppressed sequence is asked.
- If an already designed oligonucleotide is, after a new RefSeq release, more specific to a new RefSeq sequence in the current RefSeq version than it was for the originally targeted sequence, it has to be detected. The detection of false positive oligonucleotides is the purpose of the scripts described below. The specificity analysis will be performed using the SOL algorithm, to keep the same design criteria and is done between the oligonucleotide set from the n-1 Refseq version and the new sequences of the n Refseq version.

You can update oligonucleotides specificity either with the online interface ("Check oligo design specificity") or by using the SOL result files ("Insertion of the specificity analysis result files").

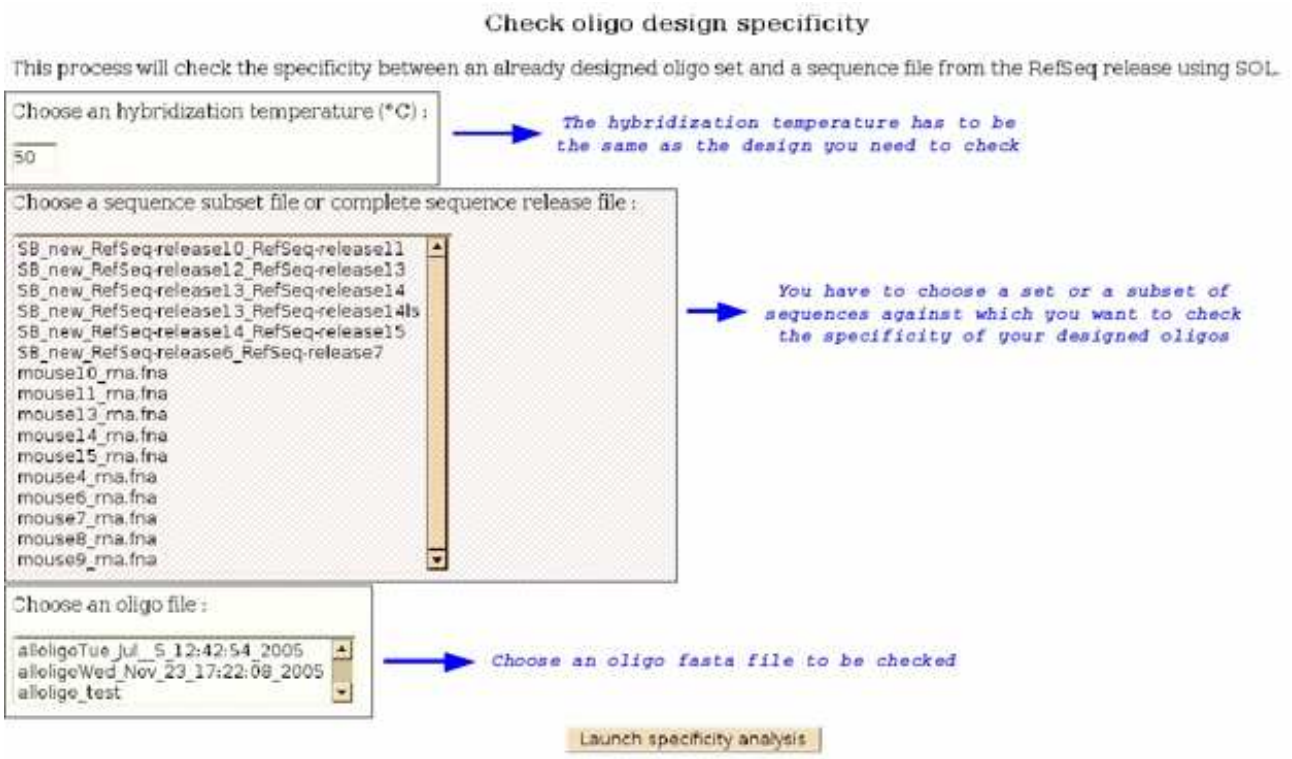


Figure 6: The web interface of the oligo design specificity check. This part displays the lists of files to be chosen to test oligonucleotide specificity. Arrows in blue describe the process to follow to launch the specificity analysis. See text for details.

The specificity analysis can be easily performed using the web interface (Figure 6). To perform this specificity check, you need to choose the hybridisation temperature, it has to be the same than the one selected from the set you want to check. Then you need to choose the sequence set or the subset of sequence against which you want to launch your specificity design. If you want to test your oligonucleotides against the new sequences of the 13th RefSeq release for example, you need to select the "SB_new_RefSeq-release12_RefSeq-release13" subset. Finally, you have to choose the fasta oligonucleotide file containing the oligonucleotide sequences to check. The formamide percentage is set to zero by default. The script "oligo_on_newseq.pl" is launched, format the parameters for the SOL algorithm. Then the "analyse_mysql_spe.pl" script launches SOL for the specificity analysis using the "feedback_nspe.pl" script to prevent the server connection to close and once the analysis is done update the SOL database.

You can also perform a direct update of the database with the result files coming from the specificity analysis generated by the SOL algorithm (see section 3.3). Upload a "...tar.gz" compressed file containing the SOL result files. When the script "validate_spe_mysql.pl" is launched, the compressed result files are saved in the temporary space defined in the "webini.pm" file (see section 1.5). Once uncompressed, the files are parsed and the oligonucleotides are updated for their specificity into the database using the script "analyse_mysql_spe.pl".

3.3The design interface: "SOL oligo design"

3.3.1 Work on result files independently from the SOL algorithm and insert new oligonucleotides in the database

You can insert the oligonucleotides designed by the SOL algorithm without using the web interface (warmly advised for large designs that contain more than 1000 oligonucleotides). SOL output

creates 10 result files depending on temperature. Build a gzipped tar file with these different temperature result files. These files can be compressed (using tar) in a directory or not, the Perl script can manage both. Upload this tar file in the "Database insertion of designed oligos" part of the form. This form launches the script "validate_result_mysql.pl".

The oligonucleotide result files from the SOL algorithm contain the RefSeq sequence identifiers followed or not, if no oligonucleotide was designed, by the oligonucleotide designed parameters and the oligonucleotide sequence. The same oligonucleotide can be designed more than once if the hybridisation temperature suits it. This means it can be found in more than one result file and will have to be tracked and not to be inserted redundantly in the database. Here is an example of a SOL result file (Figure 7):

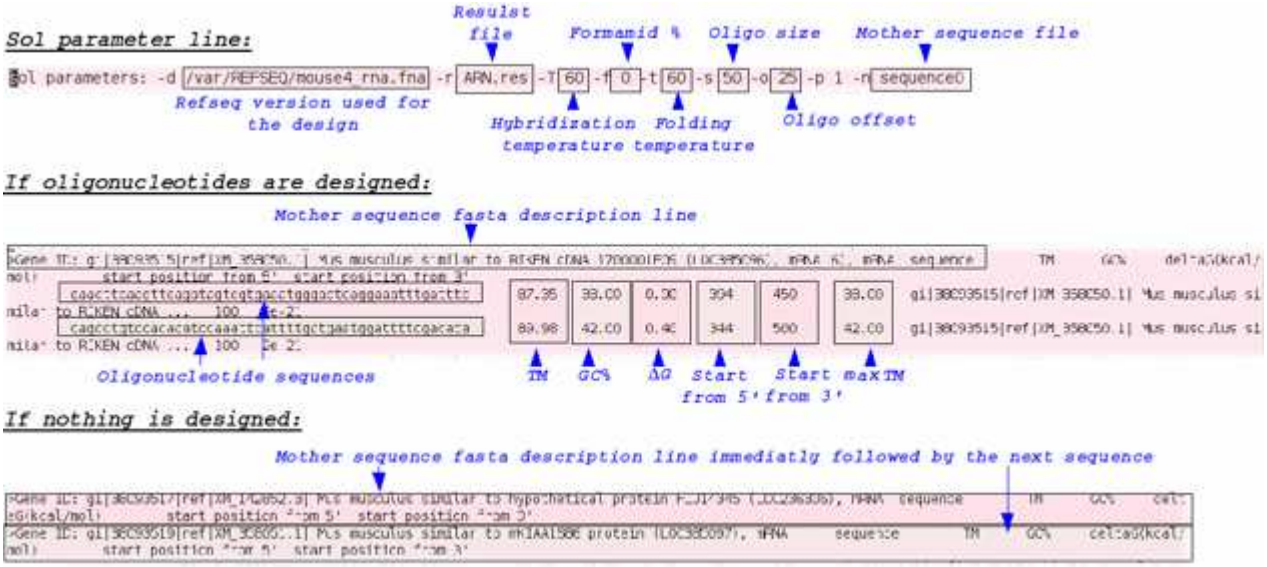


Figure 7: Example of the SOL algorithm result files. The lines from SOL output files are underline in light pink. The information given in blue explains how these output files are built. See text for details.

To avoid multiple insertion of the same oligonucleotide in the database, a unique key was created in the SQL code composed of the sequence identifier and the oligonucleotide sequence. So, an association between one RefSeq sequence and one oligonucleotide can be inserted only once in the database, and it will be inserted at its first hybridisation temperature. When the script "parse_mysql_result.pl" is launched, the result files are parsed to retrieve each sequence identifiers, the oligonucleotide parameters and the oligonucleotide sequence. The retrieved sequence identifiers are compared to those stored in the database and if the association between the sequence and the oligonucleotide does not exist, the new oligonucleotide sequence is inserted into the table "Oligo". In the same time, the oligonucleotide is written in a fasta file called alloligo_date so that it can be re-used later in the SOL algorithm for the specificity check (see section 3.2.4).

3.3.2 Launch SOL from the interface and insert the oligonucleotides in the database: "Oligo design followed by database insertion"

You can launch the SOL design algorithm from the web interface in the section "Oligo design followed by database insertion" of the form (Figure 8). This is not advised for large designs (more than 100 oligonucleotides) because it does not work in batch mode and needs a live web page to work: If the web page is closed, the design-insertion pipeline is broken.

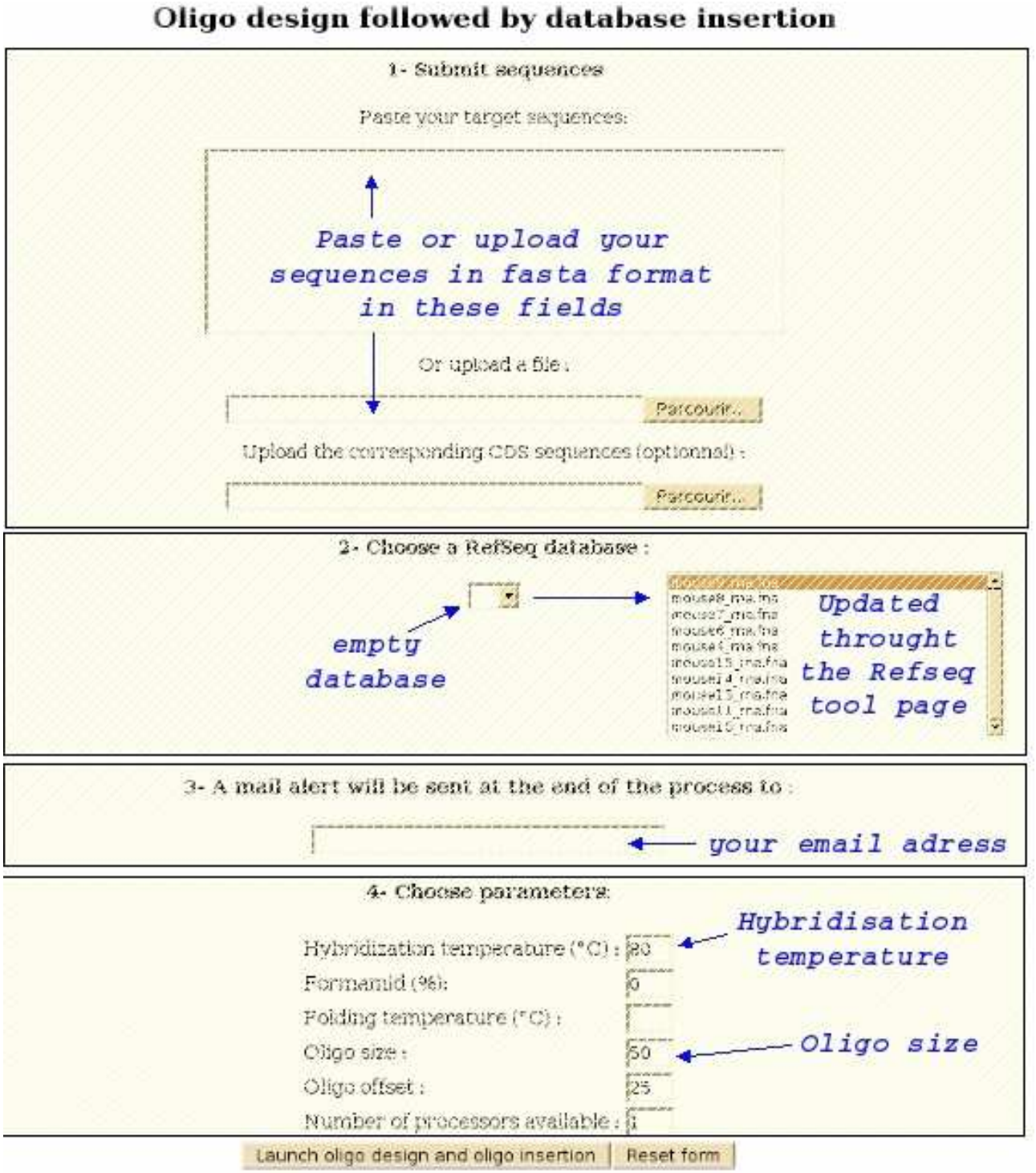


Figure 8: The "oligo design followed by database insertion" part of the design interface. This form helps to launch the SOL design algorithm. In blue are indicated what the user must fill to ensure the correct work of the SOL algorithm. In section 2 of this form, if no RefSeq database has been loaded into the SOL database nothing is displayed (empty database), which is not the case if the database has been updated (updated through the RefSeq tool page).

The first part of the form (Figure 8-1) is dedicated to the target sequences from which you want to design oligonucleotides. They can be pasted in a text field or uploaded from a file. In the second part of the form (Figure 8-2), you have to choose the RefSeq sequence version you want to use for the design. If you have never used the RefSeq tools described in the section 3.2 of this documentation, nothing can be chosen, you have to go to the "RefSeq utilities for SOL" section first, to update the database and retrieve the RefSeq sequences so that a set of formatted RefSeq sequences can be available for the SOL algorithm. In the third part of the form (Figure 8-3), a text

field has to be filled with an email address so that you can be alerted when the design and the insertion are achieved. In the last part of the form (Figure 8-4), the design parameters like the hybridisation temperature and the oligonucleotide size have to be filled in. The script "validate_form_mysql.pl" formats the parameters and files for the SOL algorithm and the script "parse_mysql_result.pl" launches the SOL algorithm using the "feedback.pl" script to maintain the web connection active. When the job is completed the script "sendmail.pl" sends a mail to the address entered on the form. Then, the result files follow the same process as described in the previous chapter (see section 3.3.1).

3.4Query the SOL database: "Retrieve oligo from the SOL database"

To obtain information from the database (Figure 9), you can submit either a list of accession numbers for the sequences you want an oligonucleotides for, or a fasta sequence file containing these sequences, the script will parse the fasta description line to get the sequence accession number for you. Next you choose a hybridisation temperature and a formamide percentage and launch your query.

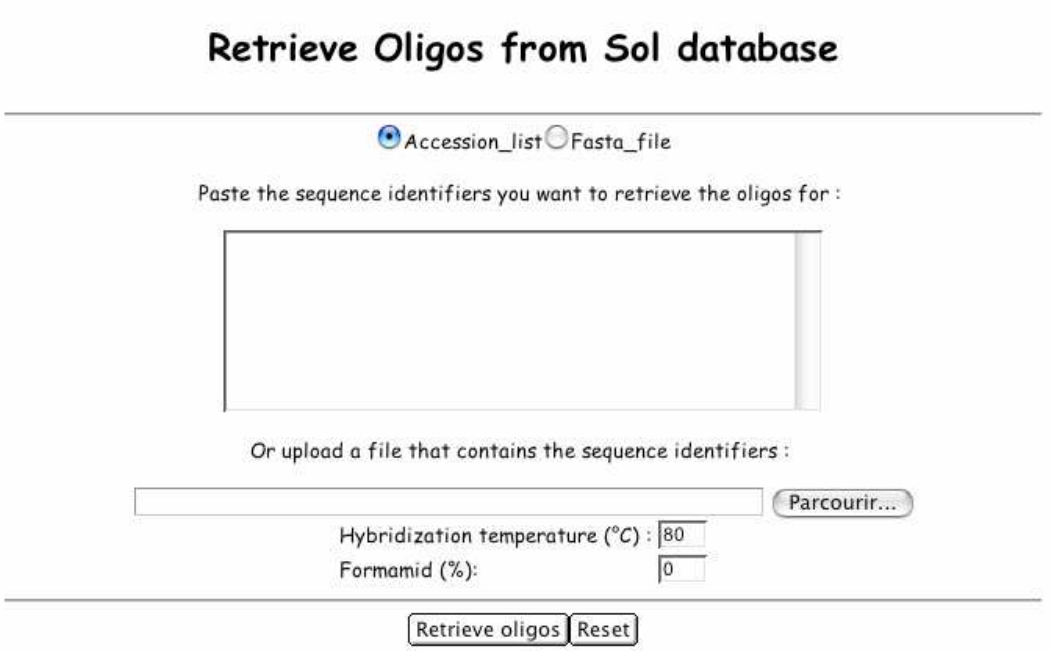


Figure 9: The SOL database query interface. This form allows retrieving oligonucleotide informations store in the database using gene identifier only. These identifiers can be loaded from the text field by copy/paste or from a fasta file. The parameters needed for oligonucleotide specificity are the hybridisation temperature and the formamide percentage.

The first part of the web page retrieved is a summary of the query posted containing the asked parameters and sequences (Figure 10). The script "retrieve_mysql_oligo.pl" searches the database for each entered sequences. If it finds a hit, the script output "screen_oligo.pl" makes the difference between a target sequence without any designed oligonucleotide and a target sequence for which oligonucleotides have been designed. If the sequence is not found in the database, the user will be notified. If the sequence is found, but without any designed oligonucleotides, the script gives you the status of your sequence (Figure 10). We consider that there is no use to give you oligonucleotides for obsolete sequences. To achieve this, the script will check the status of each sequence. For sequences where oligonucleotides have been found, you get the number of found designed oligonucleotides for each sequence and among them, those that fit the hybridisation parameters (Figure 10).

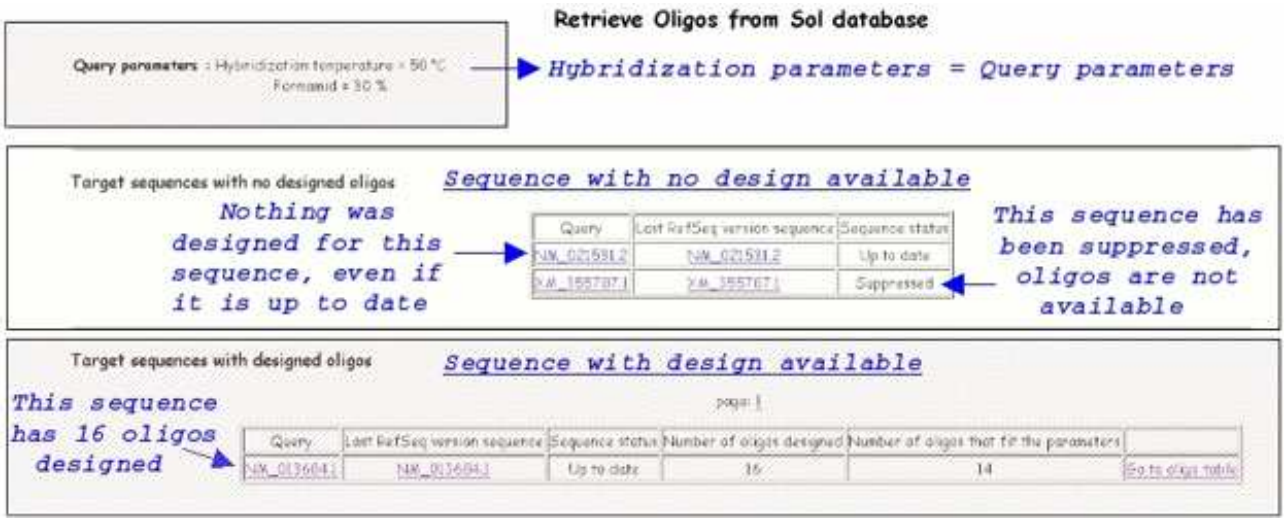


Figure 10: The oligo search output. This screenshot shows the web page retrieved when an oligo search is launched on the SOL database. In blue information are displayed on how to interpret the results found.

The second part of the result page is dedicated to the found oligonucleotides (Figure 11). These oligonucleotides are ordered according to mother sequence. Main oligonucleotide parameters are summarized in an array including hybridisation temperature, formamide percentage, TM, Delta G, GC percentage, and 5' and 3' position. The last column contains the oligonucleotide sequence. If the oligonucleotide is not considered as specific anymore, it will not be retrieved during the query.

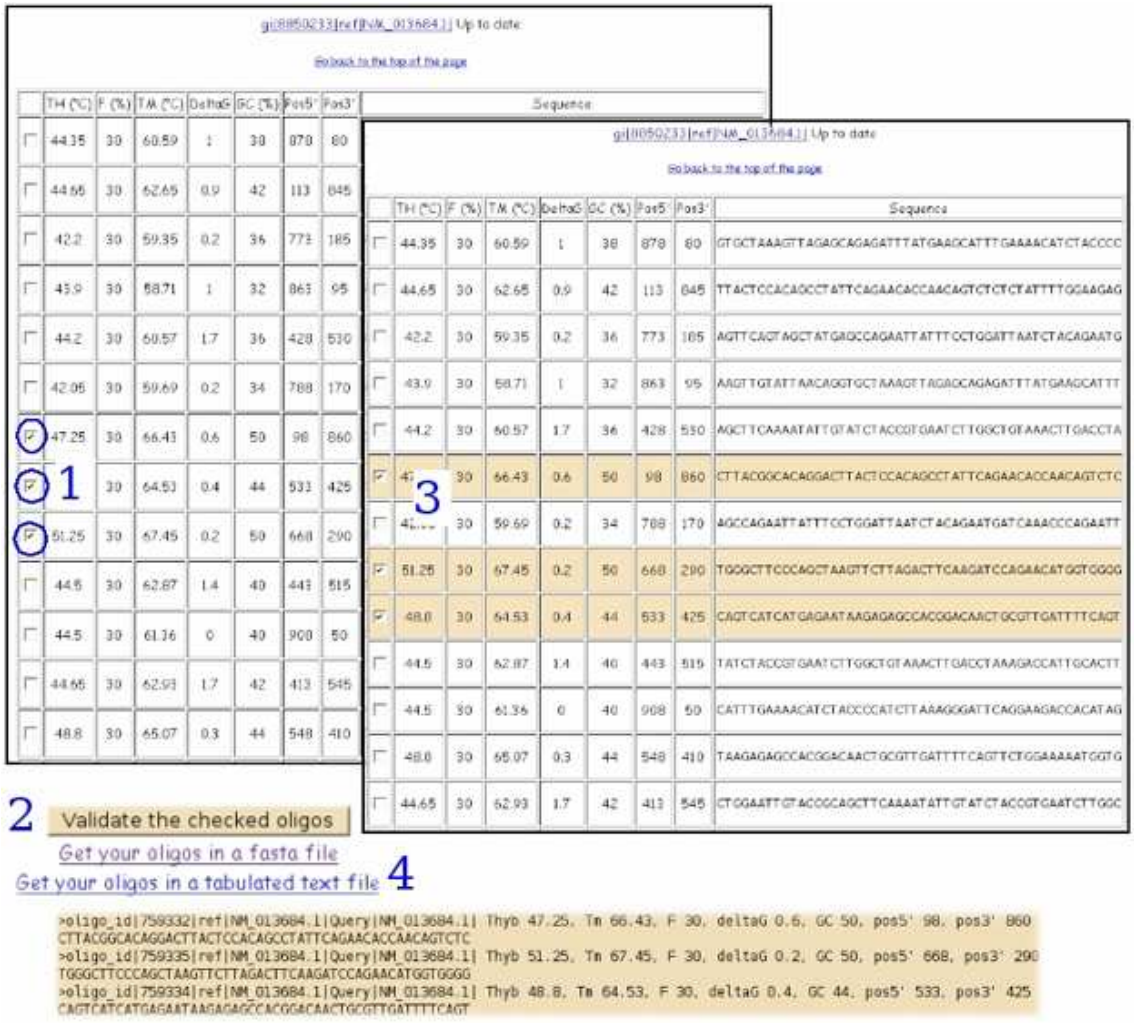


Figure 11: Oligonucleotide output from the database query. The list of all the oligonucleotides found corresponding to the submitted criteria is displayed. It is possible to select some of them (1), validate this

selection (2) and these oligonucleotides are highlighted on the web page (3) and can be retrieved in a fasta file (4).

If you want to select oligonucleotides, click the first column box (Figure 11-1), then validate your list at the bottom of the page (Figure 11-2), the validated oligonucleotide lines are then coloured in orange (Figure 11-3). You can always add oligonucleotides to your selection, but be careful not to forget the list validation. Then you can export your selection as a fasta file using the "get_data_fasta.pl" script or in a tabulated text file using the "get_data_tabulated.pl" script (Figure 11-4).

4 Adding other organisms to the database and to the web server interface

4.1 Modification to make into the database

Open a mysql console (where SOL user, password and database are SOL, pass and SOL respectively):

```
$ mysql -u SOL -ppass SOL
```

Type these lines to add a new organism into the table "Species" (see section 2.2). You can use whatever identifier you want for the organism but to avoid any duplication of the organism identifier we recommend using the taxon identifier from the NCBI website (http://www.ncbi.nlm.nih.gov.csulib.
ctstateu.edu/Taxonomy/). Be careful all the modification applied on the following section of the documentation has been tested only with organisms from the vertebrate-mammalian group:

```
mysql> insert into Species (name) values ('What_you_want_to_add');
```

4.2 Modification of the web interface

Open the script "index_mysql_refseq.pl" using a text editor and add the species you want in the section "RefSeq database Update", see the example on Figure 12.

```
print $req->p('1-Choose the species you want to retrieve the sequence for:'),
  $req->p(radio_group(-name=>'sp',-default=>'Mus musculus',
  -values=>['Mus musculus','Homo sapiens','NEW SPECIES'],
  -linebreak=>'true'));
```

Save your modification and reload the web interface, the new organism has been added (Figure 12).

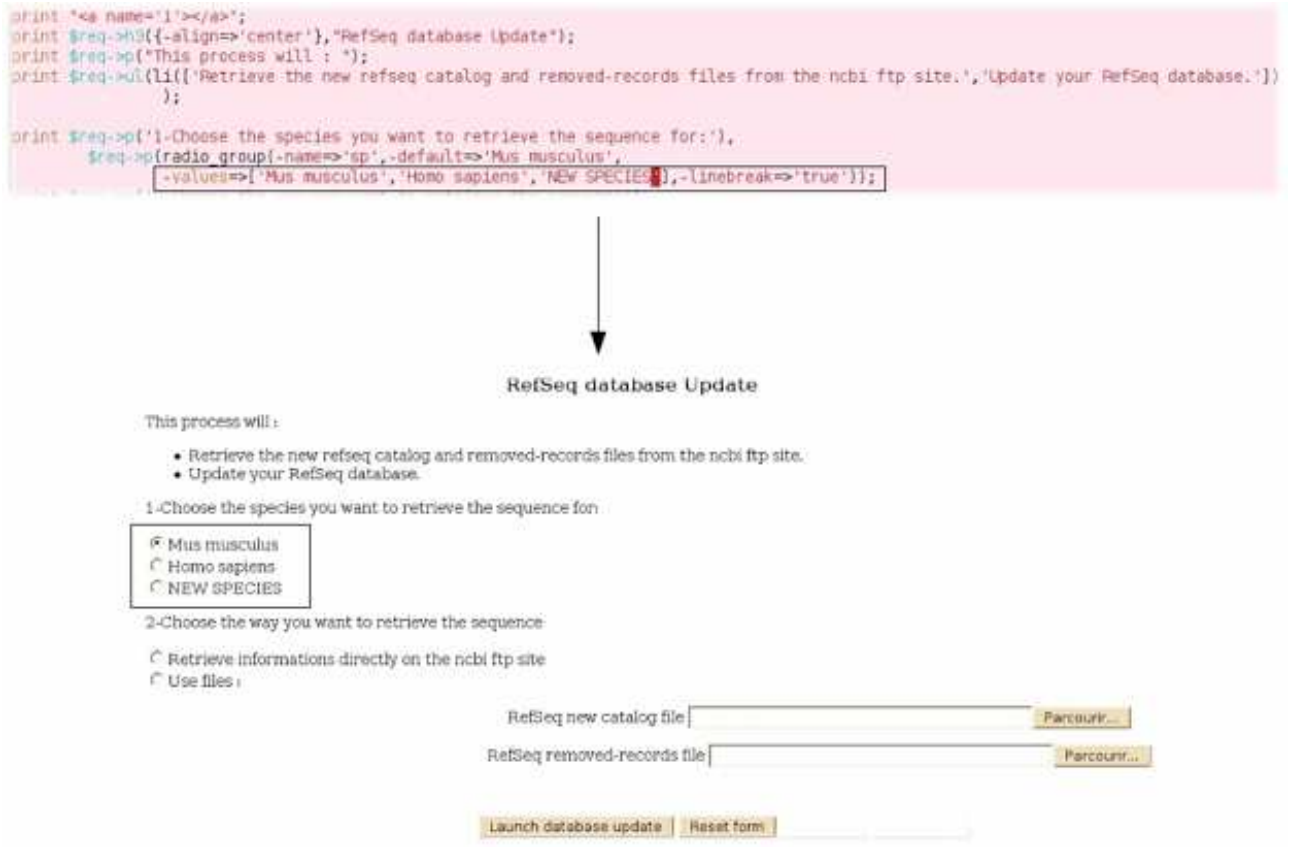


Figure 12: This figure displays the modification to make to the script "index_mysql_refseq.pl" to use a new organism using the SOL interface.

In the same script, go to the section "RefSeq sequence Update" and type for example (Figure 13):

```
print $req->p({-align=>'center'},
  $req->p('Choose the species you want to retrieve the sequence for:'),
  $req->p(radio_group(-name=>'sp',-default=>'Mus musculus',
  -values=>['Mus musculus','Homo sapiens','NEW SPECIES'],
  -linebreak=>'true')),
```

Save your modification and reload the web interface, the new organism has been added (Figure 13).

Figure 13: This figure displays the modification to make to the script "index_mysql_refseq.pl" to use a new organism using the SOL interface.

In the "retrieve_mysql_sequence.pl" script, at the beginning, add the new organism name used in the Refseq fasta files. This name needs to be exactly the same so that the sequence files can be parsed properly without missing sequences and choose a nickname for the interface use. Do not forget to save your modifications (Figure 14).

```
if($req->param('sp') eq "Mus musculus")
{
        $sp="Mus musculus";
        $sp_nickname="mouse";
}
elsif($req->param('sp') eq "Homo sapiens")
{
        $sp="Homo sapiens";
        $sp_nickname="human";
}
else
{
        $sp="New Species";
        $sp_nickname="New";

}
```

Figure 14: Example of the modification to perform in the "retrieve_mysql_sequence.pl" script.

Repeat this in the script "insert_mysql_refseq.pl", at the beginning. There, you do not need to add a nickname. Save your modifications (Figure 15).

```
if($req->param('sp') eq "Mus musculus")
{
        $sp="Mus musculus";
}
elsif($req->param('sp') eq "Homo sapiens")
{
        $sp="Homo sapiens";
}
else
{
        $sp="New Species";
}
```

Figure 15: Example of the modification to perform in the "insert_mysql_refseq.pl" script.

5 About this document

This document was written by Sophie Lemoine (slemoine@biologie.ens.fr), Stephane Le Crom (lecrom@biologie.ens.fr) and Aniss Bendjoudi (aniss.bendjoudi@bde.espci.fr)

It was last modified the 23/05/08 by Alexandre van Miltenburg (alexandre@vanmiltenburg.fr)

Please send general questions and feedback as well as errors report - either in the content or the presentation - regarding this document to marie-claude.potier@espci.fr

If you are able to provide suggested text, either to replace existing incorrect or unclear material, or additional text to supplement what's already available, we would appreciate the contribution.

Feel free to modify this document (but don’t forget to keep track of your modifications)