DEODAS 0.1.2

DEgenerate Oligonucleotide Design & Analysis System

Karl T. Diedrich

Copyright © 2000 Karl T. Diedrich

Published by Karl T. Diedrich
The DEODAS manual is available from the DEODAS web site
http://deodas.sourceforge.net/

Permission is granted to make and distribute verbatim copies of this manual provided the copyright notice and this permission notice are preserved on all copies.

Permission is granted to copy and distribute modified versions of this manual under the conditions for verbatim copying, provided also that the sections entitled "Copying" and "GNU General Public License" are included exactly as in the original, and provided that the entire resulting derived work is distributed under the terms of a permission notice identical to this one.

Permission is granted to copy and distribute translations of this manual into another language, under the above conditions for modified versions, except that this permission notice may be stated in a translation approved by the Free Software Foundation.

Abstract

A GNU/Linux-based system, called the DEgenerate Oligonucleotide Design & Analysis System (DEODAS), for designing and electronically analyzing consensus-degenerate oligonucleotides is being developed by integrating published software tools Clustalw, CODEHOP, and EMBOSS (see Chapter 6 [Rose 1998 & Thompson 1994], page 19). DEODAS also adds new functionality to oligonucelotide design. Related protein sequences are input into the system. Clustalw produces multiple sequence alignments and phylogenetic tree files. Subgroups are formed based on the phylogenetic tree. The subgroups are realigned and degenerate oligonucleotides are produced by CODEHOP. The oligonucleotides are screened against Genbank sequence databases by `fuzznuc` from EMBOSS. DEODAS integrates these software tools to automate the design and screening of oligonucleotides in batch. This greatly decreases the amount of interactive time required to design and screen probes. Output detailing the designed oligonucleotides is organized in searchable databases that are used to examine and select probes and PCR primers for synthesis and laboratory testing.

1 Introduction

With the fast expansion of DNA and protein databases in recent years, the design of oligonucleotides based on these databases has become important for molecular biologists. In particular the databases of gene seqeunces can be used to design degenerate oligonucleotides based on groups of related genes (or families) in the databases. The degenerate oligonucelotides are commonly used to discover new family members in experimental conditions.

The designed oligonucleotides can be easily synthesized and used as hybridization probes as well as Polymerase Chain Reaction (PCR) primers. However, designing oligonucleotides is not easy to do manually especially when large number of oligonucleotides for many gene families are needed. For example, DNA microarrays which can detect thousands of genes in parallel, need thousands of oligonucleotides to attach on microarry chips.

To help design oligonucleotides many computer programmers have been writing oligonucleotide design programs. Some of these programs can be used on the internet and are provided without cost. However, one of the biggest obstacles to use these programs is a lack of integration. To design oligonucleotides and examine homology to known sequences requires the use of multiple programs running on different computer platforms, and extensive manual reformatting of data between applications.

This has lead to the development of the DEgenerate Oligonucleotide Design and Analysis System (DEODAS). DEODAS integrates existing programs and automates the process of oligonucleotide design for multiple protein families. Oligonucleotides for protein families can be designed in large batches. This allows the researcher to set up the program to design oligonucleotdes for any number of protein families and let the computer do the work by itself. This greatly cuts down on interactive work time.

In addition DEODAS adds features to the design process to aid in producing and selecting high quality oligonucleotides. The program divides dissimilar sequences into subfamilies and produces oligonucleotides targeted to the specific subfamily instead of the whole, making more specifically targeted oligonucleotides. The designed oligonucleotides are automatically screened against gene databases for homology. Through this screening process the oligonucleotides are comfired against the gene sequences they were designed from and checked for the possibility of cross-hybridization to unrelated sequences. The design and screening results are automatically stored in a searchable database. This database can be analyzed interactively by the researcher giving the researcher control over which oligonucleotides are selected for use.

2 Installation

DEODAS is made up of a number of programs. Some programs are prerequisites commonly found in Linux systems and others are installed with DEODAS. Most DEODAS programs have been packaged in RedHat Package Management (RPM) format for easy installation. DEODAS also needs Genbank flat files for analyzing designed oligonucleotides. DEODAS stores its results with the Postgresql database management system (DBMS). The database must be initialized and users must be enabled before using DEODAS. These topics are covered in detail in the sections below.

2.1 Sources

DEODAS is available in source code and some parts are in binary RPM packages from the web site at http://deodas.sourceforge.net/ .

DEODAS is made of several parts. The components come as source code in the compressed archive '`DEODAS_src.tar.gz`'. One component, EMBOSS, does not come with DEODAS because it is in rapid development. EMBOSS is down-loadable from http://www.uk.embnet.org/Software/EMBOSS/ . The archive decompresses into the directory '`DEODAS_src/`' and the subdirectories '`DEODAS/`', '`support/`' and '`htdocs/`'. DEODAS contains programs written specifically for DEODAS. Support contains programs by other authors collected here for convenience to users. Htdocs contains a copy of the DEODAS web site. These directories have further subdirectories for each component. The directories are arranged this way so that '`DEODAS/`' can be kept separately on Current Version System (CVS) and programs can be easily plugged in and out in future versions of DEODAS. The licensing of each program is covered separately in its subdirectory. The program directories are outlined below.

DEODAS was developed on Intel Pentium processors with RedHat Linux 6.2. Most DEODAS programs also come as source and binary RPMs. DEODAS should also work on other Linux and POSIX or UNIX type systems that have the required libraries and programs described below. The sources and packages are outlined:

DEODAS uses a modified version of the program `mablock`. The modified source file is in this distribution and is in '`DEODAS_src/DEODAS/extra/`'. The `mablock` program normally checks the file type after opening a file but this causes a SEGMENTATION FAULT when using Glibc 2.1 or newer libraries when opening `clustalw` alignment files. In the BLIMPS '`mablock.c`' file, seq_type was set to AA_SEQ.

```
mablock.c:
line 186: seqtype = AA_SEQ;
```

The original '`mablock.c`' file is included with the extension `.orig`.

Directory: ‘DEODAS/’

Directory	Programs/Function	RPM package
‘cleanfasta/’	cleanfasta removes long descriptions from FASTA files	‘cleanfasta-*.*.*.i386.rpm’
‘deodas/’	deodas.py is the main interface, deodasdesigner.py designs oligos	‘deodas-*.*.*.i386.rpm’
‘deodasquery/’	deodasquery is for analysis of design results	‘deodasquery-*.*.*.i386.rpm’
‘deodas-doc/’	Installation guide, manuals and tutorials	‘deodas-doc-*.*.*.i386.rpm’
‘extra/’	SQL scripts and modified files of supporting programs for developers	none
‘libdbg/’	C function library with debugging for splittree	‘libdbg-*.*.*.i386.rpm’
‘libodbccppwrap/’	C++ ODBC class used in deodasquery	‘libodbccppwrap-*.*.*.i386.rpm’
‘libstr/’	C function library for specific string manipulation in splittree	‘libstr-*.*.*.i386.rpm’
‘splittree/’	splittree divides FASTA sequences into subgroups based on dissimilarity	‘splittree-*.*.*.i386.rpm’

Directory: ‘support/’

Directory	Programs/Function	RPM package
‘blimps-3.3.2/’	mablock finds conserved blocks, codehop designs oligos	none
‘clustalw-1.8/’	clustalw aligns multiple sequences	‘clustalw-1.8.i386.rpm’
‘htmlize-codehop/’	htmlize-codehop converts codehop oligo output into HTML	‘htmlize-codehop-*.*.*.i386.rpm’

2.2 Prerequisites

DEODAS uses a number of system files that usually come with Linux systems. They should be installed before DEODAS. All should come as RPM (RedHat) or DEB (Debian)

packages on most Linux systems. The graphical interfaces for DEODAS are made with the GTK and Gnome toolkits. If compiling DEODAS the development libraries are needed. If installing from binary RPMs the Gnome run-time libraries are enough; if the Gnome desktop can run these are installed. Other required programs are:

- gtkmm, available at http://gtkmm.sourceforge.net/
- libsigc++, available at http://libsigc.sourceforge.net/
- python
- pygtk
- pygnome
- postgresql, get the newest version from http://www.postgresql.org/
- postgresql-server
- postgresql-python
- postgresql-odbc
- enscript, needed for `deodasquery` printing
- expect, used to automate Genbank flat file download and installation

2.3 Compiling

Most DEODAS programs come with GNU `configure` scripts to automate compiling and installation. To build and install the programs in the subdirectories '`cleanfasta/`', '`deodasquery/`', '`htmlize-codehop/`', '`libdbg/`', '`libodbccppwrap/`', '`libstr/`', and '`splittree/`' run: `./configure`, `make`, and change to `root` to run `make install`. EMBOSS compiles in the same fashion but it is necessary to leave EMBOSS where it was compiled. The EMBOSS programs use definition files in '`EMBOSS/emboss/acd/`'. These commands should build and install the programs in those directories. The subdirectories contain additional installation information if there is a problem.

The '`deodas/`' directory contain Python and Expect scripts. These install with the command `make install` as `root` user. The installation script assumes the location of Python is '`/usr/lib/python1.5`'. The '`Makefile`' under '`deodas/src`' will need to be altered if Python is in a different location.

The programs in '`clustallw-1.8/`' compile and install with `make` and as `root` run `make install`

The directory '`blimps-3.3.2/`' must be placed in the location it is going to run from. Place the directory in a location like '`/usr/local`' and set the environment variable *`BLIMPS_DIR=/usr/local/blimps-3.3.2/`*. This can be done several ways. To make it permanent for all users as `root` open the file '`/etc/bashrc`' and enter the lines *`BLIMPS_DIR=/usr/local/blimps-3.3.2 export BLIMPS_DIR`*. Start a new x-terminal to activate the changes. Change into the directory '`blimps-3.3.2/blimps`' and compile for Linux with the command `make -f Makefile.Linux`.

The '`bin/`' directory of blimps-3.3.2 needs to be on the executable path. as `root` enter the a line like '`export PATH=$PATH:/usr/local/blimps-3.3.2/bin`' in the file '`/etc/profile`'.

Test the installation by running `mablock` and `codehop`. Just cancel with ⟨Crtl⟩c after getting some output to make sure they run.

2.4 RPM Install

The RPM packages install with the command `rpm -i` *RPM name*, all the packages can be put on the command line together or RPMs can be installed with the graphical package manager `gnorpm`. The only complication is the order. Have the prerequisites installed first (see Section 2.2 [Prerequisites], page 4). Then install the DEODAS libraries '`libdbg-*.*.*.i386.rpm`', '`lidodbccppwrap-*.*.*.i386.rpm`', '`libstr-*.*.*.i386.rpm`' first. Then install the other packages: '`splittree-*.*.*.i386.rpm`', '`clustalw-1.8.i386.rpm`', '`cleanfasta-*.*.*.i386.rpm`', '`htmlize-codehop`', '`deodas-*.*.*.i386.rpm`', and '`deodasquery-*.*.*.i386.rpm`'.

The programs EMBOSS and BLIMPS have to be compiled to install (see Section 2.3 [Compiling], page 5).

2.5 Genbank installation

Genbank flat files are available by anonymous `ftp` at `ftp://ncbi.nlm.nih.gov` in the directory '`genbank/`'. An Expect script is included for automating the download and installation of Genbank flat files. It will be installed as '`/usr/bin/ftp.genbank.exp`'. Three parts of this file need to be modified. The parts are each labeled with #TODO. So after opening this script with a text editor search for #TODO. On line 71 enter the directory to store the Genbank flat files. The user running the script must have write access to this directory. It is more secure for a normal user to run this script than the `root`. On line 90 enter a user e-mail address for an anonymous ftp password. On line 101 enter the Genbank files needed for this installation.

The script can be automatically ran by `cron`, a program that runs programs periodically. To set this up a user should run the command `crontab -e`. This brings up the `vi` text editor. Enter a line with *minute hour day of month* * * `ftp.genbank.exp`. An example could be:

```
15 1 1 * * ftp.genbank.exp
```

This would be to download Genbank on the first of each month at 1:15 am. Please make up your own numbers or we'll all be accesses the server at the same time. Then write and quit the `vi` text editor with `:wq`.

2.6 Postgresql

To install Postgresql database management system (DBMS) get the latest version from `http://www.postgresql.org/` . They come as sources and RPMs. DEODAS 0.1 was written using Postgresql 7.02. Get the RPMs if your system can install RPM packages. Install them with `rpm -i` *package-name-*.*.*.i386.rpm* or with a graphical installer like `gnorpm`. After

installing the packages initialize the system from the `root` user by running the command `/etc/rc.d/init.d/postgresql start`.

The Postgresql DBMS needs to start when the computer starts. The K-Desktop Environment (KDE) contains a nice program for this called `ksysv`. Run `ksysv` as root user the drag `postgresql` to run level 5. This creates a symbolic link from '`/etc/rc.d/init.d/postgresql`' to '`/etc/rc.d/rc5.d/`' that automatically starts the Postgresql DBMS when the computer starts up and enters run level 5.

The Postgresql is managed by the user `postgres`. From `root` change to user `postgres` with the command `su postgres`. As this user enable system users to use the DBMS run the command `createuser` *user-name*. Then answer the questions at the prompt. Postgresql comes with an extensive manual and has more information at its home page. If the user is enabled with Postgresql, DEODAS can handle database creation and connection through the graphical interface.

3 How it works

In this chapter what DEODAS is doing will be covered step-by-step in the order the actions occur. Please have the program `deodas.py` running while reading this. The `deodas.py` interface was designed to be as self explanatory as possible.

All files ending '`.py`' are Python scripts. These are ran by the Python interpreter. If there is any problem starting these scripts make sure the '`python`' interpreter is on the executable path.

3.1 Input files

DEODAS starts with collections of related protein sequences, or a protein family, each in their own FASTA format file. Related proteins can be found by starting with a base sequence and conducting a BLASTP search (see Chapter 6 [Altschul 1997], page 19). The BLASTP search compares the base sequence with proteins sequences reverse translated from nucleic acid databases (see Chapter 6 [Henikoff 1990], page 19). Due to automated sequencing the number of nucleic acid sequences far exceeds the number of directly sequenced proteins. More than one FASTA file and protein family can be processed at once but they will receive the same settings (seen later). Make sure all protein sequences for a family are in a FASTA format file together (see Chapter 4 [Tutorial], page 16). All FASTA files to be processed are placed in a single directory.

BLASTP search results are the recommended input for DEODAS. DEODAS has been extensively tested with BLASTP protein input. In these file the second sequence is the closest relative to the top sequence and they diverge as the sequences go down the file. If the sequences are entered in random order in the input file the `splittree` program may not group subfamilies correctly if close relatives are far apart in the FASTA file but this has not been well tested.

A file extension changing utility program, `extension.py` is included with DEODAS. It changes all file extension in a directory to another. It runs with arguments `extension.py` *file-extension-from file-extension-to*. For example if there are several '`*.fas`' file in a directory run the command `extension.py fas fasta`. The `extension.py` command also runs interactively. Run `extension.py` and enter the file extensions at the command prompts.

3.2 Interface

The DEODAS interface program `deodas.py` selects a directory to run the designer process, `deodasdesigner.py` on. Every '`*.fasta`' file is operated on by `deodasdesigner.py`, other file extensions are not recognized even if it is a FASTA file. The `deodasdesigner.py` runs as a separate process because the design and analysis process can take over a day if many sequences are involved. The interface `deodas.py` can be turned off once a design process is started or can be used to start a new process in another directory.

See figure 1 next page, DEODAS main interface.

3.3 Figure 1 DEODAS main interface `deodas.py`

3.4 Codon Usage table

Settings for DEODAS include, the ⟨Codon usage table⟩ tool bar item, seen on the interface. The codon usage table is used by `codehop` to back-translate amino-acid sequences into nucleic-acid sequences (see Chapter 6 [Nakamura 1997], page 19). Codon usage tables are kept in `$BLIMPS_DIR'/docs'`. Additional codon usage tables are available from `http://blocks.fhcrc.org/blocks/help/CODEHOP/codon.html` .

3.5 Maximum subfamily dissimilarity

The ⟨Maximum subfamily dissimilarity⟩ range is used to divide the the FASTA protein family files into sub-families based on their dissimilarity. A family tree of each protein family is create by `clustalw` and `splittree` reads the tree and looks for sections branching above the dissimilarity set in `deodas.py`. Branches of the tree exceeding the dissimilarity setting are broken off into sub-families. A new FASTA file is written for each subfamily with the name *original-base-file-name.sub-number*`.fasta`. For example '`uspA.fasta`' could split into '`uspA.1.fasta`' and '`uspA.2.fasta`'. The subfamily splitting is needed because if the sequences in the original '`*.fasta`' file are too diverse no oligonucleotides can be produced. It may be necessary to try designing oligonuceotides for a family with a few different dissimilarity values.

3.6 Minimum oligonucleotide length

The ⟨Minimum oligonucleotide length⟩ range sets `deodasdesigner.py` to filter out oligonucleotides below the size from the final data. These oligos still appear in intermediary files discussed later.

3.7 Maximum mismatches

The ⟨Maximum mismatches⟩ range adjusts the mismatches allowed when searching the designed oligonucleotides against Genbank flat files. After oligonucleotides are designed each oligonucleotide over the ⟨Minimum oligonucleotide length⟩ is searched against Genbank for matches by the `fuzznuc` program from EMBOSS. This setting adjusts the mismatches allowed in this search.

3.8 Genbank setting

⟨Genbank databases⟩ entry selects the Genbank flat files that the oligonucleotides are searched against. The flat files are entered with their full path names separated by commas with no spaces. Genbank flat files are at `ftp://ncbi.nlm.nih.gov/genbank/` . An Expect script,

'`ftp.genbank.exp`', can be used to automatically download and install Genbank flat files (see Section 2.5 [Genbank installation], page 6).

3.9 Results databases

The ⟨Create a new results database⟩ entry creates a new database with the Postgresql database management system (DBMS). To use this the user must have been enabled with the Postgresql DBMS (see Section 2.5 [Genbank installation], page 6). The results of the oligonucleotide design are saved to this database. The ⟨Save results to an existing database⟩ entry sets `deodasdesigner.py` to save it's results to an existing Postgresql database. The user must be allowed to write to this database under the Postgresql DBMS's authority. If multiple users save to the same database consults the Postgresql manual.

3.10 Running

The ⟨OK⟩ button starts `deodasdesigner.py`. As mentioned before this is a separate process and the `deodas.py` interface can be used again in another directory or tuned off. Be careful not to start a process in same directory by pressing ⟨OK⟩ again or the second `deodasdesigner.py` process will overwrite the earlier process's files.

The first released version of DEODAS uses the operating system's normal process control system. The command `ps` displays the processes running from a x-terminal. This can be used to monitor the DEODAS processes. To stop `deodasdesigner.py`, run `ps` from the terminal running `deodasdesigner.py` to get the process number. The run the command `kill` *process-number*. As an alternative the K-Desktop Environment (KDE) comes with a a graphical process monitoring system, `kpm` that can be used to monitor and kill processes. It comes in the '`kde-utils`' package in many Linux systems. It also requires that KDE is installed. Turning off the terminal running `deodasdesigner.py` will stop everything if needed. Future versions of DEODAS may contain more process control.

3.11 Design & screening

The design and screening of the oligonucleotides is handled by the `deodasdesigner.py`. As before it runs in its own process. It reads all '`*.fasta`' files in the starting directory (see Section 3.1 [Input files], page 8). Then it processes each file one at a time until finished.

Next `deodasdesigner.py` creates a subdirectory with the name of the FASTA file minus the ending '`.fasta`'. For example '`uspA.fasta`' creates directory '`uspA/`'. This keeps all intermediate files generated for a FASTA file in one directory together. The settings for the program are written to a file in each sub-directory named '`settings.txt`'. The program `cleanfasta` removes any long descriptions that can interfere with many sequence analysis programs and creates a stripped copy of the FASTA file in the sub-directory.

Next `clustalw` reads the FASTA file in the sub-directory and outputs a multiple sequence alignment in the file *file-name*.'`.aln`' and a phylogenetic tree in Newick nested parenthesis

format in the file *file-name*`.dnd` (see Chapter 6 [Thompson 1994], page 19). The multiple sequence alignment is produced by aligning each pair of sequences and assigning a similarity score. The pairwise alignments are used to build the phylogenetic tree. The multiple alignment starts with the the closest aligned pair and one by one adds on the next sequence in the tree (see Chapter 6 [Lipman 1989], page 19).

The phylogenetic tree and FASTA file pair are read by `splittree`. The dissimilarity setting is used to divide the FASTA file into subfiles. `splittree` begins copying the FASTA sequences in the '`*.fasta`' into '`*.1.fasta`'. When a branch in the phylogenetic tree, '`*.dnd`', file over the dissimilarity is found a new file, '`*.2.fasta`', is created and the FASTA sequences are copied into the next subfile (see Section 3.5 [Maximum subfamily dissimilarity], page 10). Every time a branch over the set dissimilarity value is encountered a new subfamily file is created. This process continues until all the entire '`*.dnd`' tree file has been read. Splitting the sequences into subfamilies causes the oligonucleotides to be designed on more closely related sequences.

Next `clustalw` realigns the sequences in the subfamily file '`*.number.fasta`'. The oligonucleotides will be designed based on the sub-alignments.

The Blimps package's `mablock` program finds highly conserved blocks within the subfamily protein alignment (see Chapter 6 [Henikoff 1991], page 19). The blocks are ungapped regions of similarity often separated by gapped regions. Th blocks are found by two systems in parallel. On system searches for protein motifs and extending outward until similarity disappears. The other is an interative approach that trys all combinations of position and length of the blocks (see Chapter 6 [Henikoff 1995], page 19). Using two approaches in parallel double checks the reality of the block. The blocks are saved in the file '`*.blks`'. Another Blimps program, `codehop` reads the '`*.blks`' file. `codehop` forms a consensus protein sequence and back-translates into nucleic acid using the codon usage table input earlier (see Section 3.4 [Codon usage table], page 10). Oligonucleotides are designed on the back-translated nucleic-acid consensus sequence.

The oligonucleotides designed by `codehop` are consensus-degenerate sequences. The 3' region is a short, 11-12 bp, degenerate core region based on 3 or 4 highly conserved amino acids. The 5' region is a longer, 18-25 bp consensus clamp (see Chapter 6 [Rose 1998], page 19). The degenerate core helps the oligonucleotide pair with a range of related genes. The consensus core region reduces the degeneracy of the oligonucleotides population. If the degeneracy is too high a single oligonucleotide sequence may be too dilute to produce a detectable signal in hybridization reactions even if there is a match. The results from `codehop` are saved in '`*.codehop`' files. These files are translated into HTML by `htmlize-codehop` and have the name '`*.codehop.html`'. They can be viewed with any web-browser to see the consensus block sequences and the locations of the oligonucleotides against those blocks.

The next step is to electronically analyze the oligonucleotides produced by `codehop`. This is done by searching the oligonucleotide sequences against Genbank flat files with nucleic acid sequences. The `deodas.py` interface allows the selection of Genbank flat files, minimum length, and mismatches. Sequences under the minimum length are skipped by `deodasdesigner.py` at this step. The `fuzznuc` program, from EMBOSS, searches each sequence against the genes in the specified Genbank flat files, including complimentary sequences, and the results are written to files with a name *fasta-file-base-name.subfamily-number.probe-number.Genbank-flat-file*.fuzznuc (submitted for publication EMBOSS) (see

Chapter 6 [References], page 19). The data output by fuzznuc includes the matching targets' locus, description, accession number, starting position of the match, mismatches, and matching sequence. All of this data is read and formatted into the structured query language (SQL) file *fasta-files-base-name*.sql. This file is loaded into the Postgresql database selected in the interface (see Section 3.9 [Results databases], page 11). An SQL file is written instead of loading the data directly into the Postgresql database so it can be loaded into another DBMS if the laboratory already uses another DBMS.

The `deodasdesigner.py` prints

> Degenerate oligonucleotide design process finished

when it is finished. At this point the results can be analyzed interactively.

3.12 Analysis

The analysis program `deodasdesigner` can be started from the `deodas.py` interface, where it will automatically connect to the selected Postgresql database or it can be started by itself. New databases are easily connected by typing in a new database name in the ⟨Connect to database⟩ entry and pressing enter. The current database is automatically disconnected. The way to analyze the results is to search for keywords in the ⟨Search target descriptions⟩ entry. This will find oligonucleotides matching interesting targets. The next step is to search the name of an interesting oligonucleotides to find out all the targets it matched. This is to screen out oligonucleotides matching targets other than the desired one. Prescreening the designed oligonucleotides against gene databases can be used to reduce cross-hybridization (see Chapter 6 [References], page 19).

See figure 2 next page, Deodasquery interface

3.13 Figure 2 Deodasquery interface

Once an oligonucleotide is selected for use, it's name is entered into a list kept in the database by entering it's name in the ⟨Select an oligonucleotide name⟩ entry. Names that don't exist in the database can't be added to the list. So after entering a name press the ⟨List selections⟩ button to make sure the name was entered correctly. This button lists all selected sequences and information including name, sequence, block, degeneracy, melting temperature, and length (GC content will be added later). CODEHOP calculated the melting temperature based on the melting temperatures of the lest stable oligonucleotide-template pair as in the method by Rychlik (see Chapter 6 [Rychlik 1990], page 19). The text box listing the results can be printed using the ⟨Print⟩ button. Oligonucleotides can be deleted from the selected list with the ⟨Deselect an oligonucleotide⟩ entry. The `deodasquery` program includes pop-up help boxes so try running the program with results to discover more about how it works.

4 Tutorial

1. Go to the NCBI Web site with a web browser to collect protein sequences: http://www.ncbi.nlm.nih.gov/ .
2. Click proteins from the Search Box.
3. Type the target gene name ('`Bph`') into the search for box.
4. Click Go to start the search. Wait a little until the search is done.
5. Click on the name of one interesting gene which is colored blue to select it as the starting gene.
6. Copy the numeric part of the Genbank Index (GI) number.

```
Copy only the numbers of 'gi|1073448'
10173448
```

7. Goto the BLAST search on NCBI.
8. Select Basic BLAST search.
9. Select Blastp (BLAST protein search of nr)
10. Select Search by Accession or GI.
11. Paste in the GI number and start the search.
12. Click format results and the search result will show up on a browser new window.
13. Print this page to file to save the output.
14. Use the results to decide on how many sequences are similar enough to include together.
15. Go back to the starting sequence
16. Click ⟨Protein Neighbors⟩ from the Display Box.
17. Click the ⟨Display⟩ button.
18. Wait a little until the search is done and the results will show up on a new window.
19. Click on the empty square boxes of interesting genes to select them. Use the Blastp results to decide on how many to select (up to 40).
20. Click ⟨FASTA⟩ from the Display Box.
21. The search result, FASTA format protein sequences, will show up in a new window.
22. Click ⟨Save⟩ and save with the file extension '`.fasta`'. Repeat the procedure for other protein families. DEODAS can design oligos for multiple protein families as long as each family is in it's own '`*.fasta`' file.
23. Open DEODAS program by running `deodas.py &` from a x-terminal and the DEODAS user interface will show up.
24. Select the FASTA file directory. Oligonucleotides will be generated for each file ending '`.fasta`' in the directory.
25. Select a codon usage table (see Section 3.4 [Codon usage table], page 10).
26. Set the ⟨maximum subfamily dissimilarity⟩ (see Section 3.5 [Maximum subfamily dissimilarity], page 10).
27. Set the ⟨minimum oligonucleotide length⟩ (see Section 3.6 [Minimum oligonucleotide length], page 10).

28. Set the ⟨maximum mismatches⟩ (see Section 3.7 [Maximum mismatches], page 10).
29. Type in the names of the Genbank flat files to search (see Section 3.8 [Genbank setting], page 10). .
30. Either create a new results database or select an existing database (see Section 3.9 [Results databases], page 11).
31. Run the program by pressing ⟨OK⟩ (see Section 3.10 [Running], page 11). The design process can take over a day so find something else to do. Additional design processes can be started with the interface as long as they are in different directories. The x-terminal running the design process will print messages and will print '`Design process finished`' when the process is finished.
32. When the design process is finished click the ⟨query⟩ button to turn on `deodasquery` and analyze the results.
33. Search the target descriptions by putting in single keywords by typing in the keyword and pressing enter.
34. After finding an oligonucleotide matching a correct target in the Genbank databases search that oligonucleotide by name to check for incorrect matches to reduce the chance of cross-hybridization in the final oligonucleotides.
35. When an oligonucleotide is selected for synthesis copy or type the name into the ⟨Select an oligonucleotide name⟩ box and press ⟨Enter⟩.
36. All selected oligonucleotides can be listed by clicking ⟨List Selections⟩.
37. The ⟨Print⟩ button prints everything in the results box.

5 Remote use

DEODAS can be used remotely using the Virtual Network Computing (VNC) program available at http://www.uk.research.att.com/ . The VNC web-site contains information on setting up a Linux server and VNC viewers on other computers.

6 References

1. Altschul, S. and Lipman D.J. (1990) Protein database searches for multiple alignments. Proc. Natl. Acad. Sci., USA. 87, 5509-5513.
2. Altschul, S.F., Madden, T.L., et al. (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Research, 25 (17), 3389-3402.
3. Henikoff, S. and Henikoff, J.G. (1992) Amino acid substitution Matrices from protein blocks. Natl. Acad. Sci., USA. 89, 10915-10919.
4. Henikoff, S. and Henikoff, J.G. (1991) Automated assembly of protein blocks for database searching. Nucleic Acids Research, 19 (23), 6565-6572.
5. Henikoff, S. and Henikoff, J.G. (1994) Position-based sequnece weights. J. Mol. Biol. 243, 574-578.
6. Henikoff, S.,Henikoff, J.G., et al. (1994) Automated construction and graphical presentation of protein blocks from unaligned sequences. Gene. 163, GC17-GC26.
7. Henikoff, S., Wallace, J. (1990) Finding protein homologies with DNA databases. Methods in Enzymology. 183, 111-133.
8. Lipman D.J., Altschul, S.F., et al. (1989) A tool for Multiple Sequence Alignment. Proc. Natl. Acad. Sci., USA. 86, 4412-4415.
9. Mitsuhashi, Cooper, A., et al. (1994) Oligonucleotide probe design - a new approach. Nature. 367, 759-761.
10. Nakamura, Y., Gojobori T., et al. (1997) Codon Usage Tabulated from the international DNA sequence databases. Nucleic Acids Research. 25(1), 244-245.
11. Nuwaysir, E.F., Bittner, M., et al. (1999) Microarrays and toxicology: The advent of toxicogenomics. Molecular Carcinogenesis. 24, 153-159.
12. Rose, T.M., Schultz, E.R., et al. (1998) Consensus-degenerate hybrid oligonucleotide primers for amplification of distantly related sequences. Nucleic Acids Research, 26 (7), 1628-1635.
13. Rychlik, W., Spencer, W.J., et al. (1990) Optimization of the annealing temperature for DNA amplification in vitro. Nucleic Acids Research, 18(21), 6409-6412.
14. Thompson, J.D., Higgins, D.G., et al. (1994) CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. Nucleic Acids Research. 22(22), 4673-4680.
15. Weiler, J., Gausepohl, H., et al. (1997) Hybridization based DNA screening on peptide nucleic acid (PNA) oligomer arrays. Nucleic Acid Research. 25(14), 2792-2799.
16. Wilbur, W.J. and Lipman, D.J. (1983) Rapid similarity searches of nucleic acid and protein data banks. Proc. Natl. Acad. Sci., USA. 80, 726-730.

Program Index

Concept Index

Short Contents

Table of Contents