UPTEC X 01 035
JUL 2001

ISSN 1401-2138

CLAES LADENVALL

Development of algorithms for automated construction of padlock probes

Master's degree project

Molecular Biotechnology Programme
Uppsala University School of Engineering

UPTEC X 01 035 | **Date of issue 2001-07**

Author

Claes Ladenvall

Title (English)

Development of algorithms for automated construction of padlock probes

Title (Swedish)

Abstract

Padlock probes are oligonucleotide probes that can be used to detect single nucleotide variations of DNA and RNA *in situ* or in solution. They can be circularised by ligation in the presence of a perfectly matching target sequence. Design of padlock probes has so far been done manually. To automate the designing process algorithms to search for intramolecular complementarities and to determine the melting temperature, T_m, using nearest-neighbour thermodynamics, were implemented. The algorithms have been assembled in a program called makepad. Makepad can be used to generate candidate designs of padlock probes.

Keywords

Padlock probe, T_m, Smith-Waterman, intramolecular complementarity

Supervisors

Ulf Landegren
Department of Genetics and Pathology, Uppsala University

Examiner

Björn Andersson
Department of Genetics and Pathology, Uppsala University

Project name	Sponsors
Language **English**	Security
ISSN 1401-2138	Classification
Supplementary bibliographical information	Pages **23**

Biology Education Centre Biomedical Center Husargatan 3 Uppsala
Box 592 S-75124 Uppsala Tel +46 (0)18 4710000 Fax +46 (0)18 555217

Development of algorithms for automated construction of padlock probes

Claes Ladenvall

Sammanfattning

Vår mänskliga arvsmassa är uppbyggd av ett stort antal olika gener. Dessa gener är till största delen lika mellan olika individer. Små olikheter gör att vi människor kan se lite olika ut, eller klarar oss olika bra när vi blir utsatta för sjukdomar. Det finns också några genuppsättningar som direkt kan vara skadliga för den individ som bär på dem. De kan i sig leda till sjukdom på grund av att genen inte fungerar som den ska. I vissa fall kan skillnaden mellan en väl fungerande gen och en defekt vara så liten som en enda bas, den minsta byggstenen i vår arvsmassa. För att kunna titta på den här typen av små genförändringar och med säkerhet avgöra vilken typ av bas som finns i den kritiska positionen krävs mycket känsliga tekniker. En relativt ny analysmetod som är tillräckligt känslig och stabil och som dessutom möjliggör ett flertal analyser samtidigt är ”padlock probe”-tekniken.

”Padlock probe”-tekniken bygger på en typ av molekyl som kan designas för att bara fastna på en specifik genuppsättning. Om en bas inte stämmer överens kommer molekylen inte att binda in till genen och således kan man avgöra vilken bas som finns i den variabla positionen. Molekylerna som används i den här tekniken har än så länge designats manuellt. Det är ett tidskrävande arbete och det finns ett behov att ta fram algoritmer för att kunna automatisera designmomentet. Här presenteras ett antal algoritmer som tillsammans används för att bygga padlock-prober. De sätter samman varje molekyl från ett antal mindre byggstenar och verifierar sedan att den uppsättningen byggstenar ger en molekyl som kan förväntas fungera.

Examensarbete 20 p i Molekylär bioteknikprogrammet

Uppsala universitet Juli 2001

Contents

Introduction

In February 2001 the complete human genome was presented (1). Accordingly, humans only have about 30 000 - 40 000 protein coding genes. This is less than previously had been suspected, roughly twice as many as a fly or a worm. However the study of the expression pattern of our genes, the identification of their function and investigation of the genetic variation present in them will require a tremendous effort. In the human genome approximately one nucleotide per 200-1000 differ between individuals. These variations are called single nucleotide variations or single nucleotide polymorphisms (SNPs). The first draft of the human genome has revealed more than 1.4 million SNPs. In most cases these variations have no or very little effect on the phenotype, but some of them are probably involved, in combination with environmental factors, in causing complex diseases.

To be able to investigate these variations, accurate and efficient tools that simultaneously can detect genetic variation and expression patterns among a large number of genes are required. The tools must be able to distinguish a single nucleotide in a genome of three billions. Several methods have been developed to meet these needs such as the invader technique, real-time PCR and northern blot to detect specific RNA transcripts. Complex expression patterns of thousands of transcripts in various tissues can be studied with techniques such as DNA microarrays and the SAGE technique. *In situ* SNP analysis can be done with *in situ* PCR, allele specific hybridisation and primed *in situ* labelling (2). The PCR techniques are highly specific and sensitive, but cross-reactions that occur are not easily discerned. Hybridisation reactions on the other hand are usually limited in both specificity and sensitivity.

The focus of this work has been on a relatively new technique to detect SNPs quantitatively *in situ*, the padlock probes. This technique combines the advantages of PCR amplification and DNA hybridisation.

Padlock probes

Padlock probes are linear oligonucleotides of approximately 90 bases. They consist of two end segments and a connecting linker sequence. The two opposite ends are designed to hybridise to adjacent segments of a target sequence, such that the 5' and 3' ends of the probes are brought into juxtaposition, creating a double helix with a nick. The probe can then be circularised by sealing the nick with a DNA ligase (figure 1).

The simultaneous hybridisation of the two target-complementary sequences ensures a high specificity in recognising a target sequence (3). The discriminating power of padlock probes is further enhanced by the fact that the ligation reaction is strongly inhibited by mismatches at the ligation junction, especially at the 3'-ends of the hybridised probes (4). If the target nucleotide at the variable position is not complementary to the probe nucleotide, the resulting topology distortion is sufficient to prevent the ligase from creating the phosphodiester bond. It is thus possible to use padlock probes to detect single nucleotide sequence variations. The circularised molecule will bind strongly to its target, making it possible to use stringent washes when used on a solid phase, reducing the background signal from non-circularised probes.

Padlock probes are also suitable for multiplexed use. Since intramolecular ligation reactions are kinetically more favourable than intermolecular ligations**,** cross-reactions between different probes should not present a problem (5).

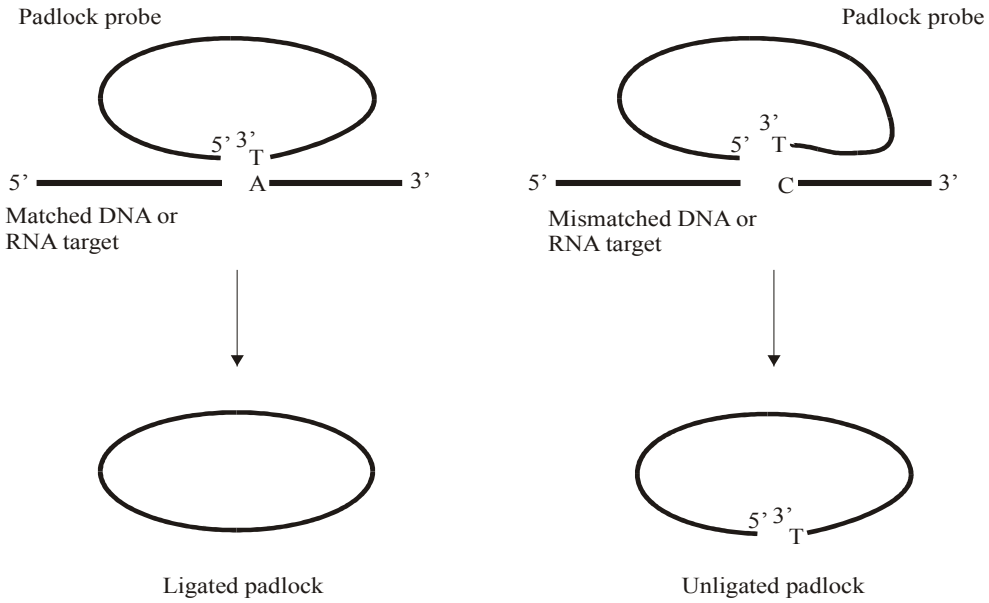

Figure 1. Hybridisation of a padlock to a matched target sequence followed by ligation with a DNA ligase will result in a circularised padlock, wrapped around its target. A mismatch at the variable position is sufficient to inhibit ligation and circularisation.

The linker segment of the padlock probe can be constructed in several ways, to accommodate a large spectrum of applications. So far the focus has been on highly specific detection of circularised probes by rolling circle amplification (RCR) (6) and PCR (7). The RCR reaction produces a long chain of complementary sequences and can be initiated by using a primer complementary to the linker segment. The RCR products can then be detected either by hybridisation of labeled probes or by incorporation of labeled nucleotides during RCR. By using a pair of linker segment specific primers, PCR can be used to specifically amplify circularised padlock probes across the ligated section.

Manual padlock probe design

Padlock probes have so far been designed manually. The end segments of the probe are designed to be complimentary to the target sequence. A set of 20mer sequences has been provided by Affymetrix and is used to build up the linker segment of the probe. The 20mers provided by Affymetrix have been selected from all possible 20mers to have similar hybridisation characteristics and minimal homology to sequences in the public databases (8). In this application these sequences are called zip codes.

In the first step of the manual probe design the end segments of the probe must be determined. They must be complementary to the target sequence and the nucleotide complementary to the nucleotide at the variable position put at the very end of the 3' end. Secondly the linker segment must be determined. Typically it is made up of three zip codes, one constant primer, one variable primer and a variable zip code (figure 2). The two primers can be used in several probes, but the zip code sequence has to be unique, since this is the part of the probe that will hybridise to the microarray in the analysis phase in order to identify the reacted probe. Each combination of sequences has to be checked for intramolecular complementarities to assure that strong internal base pairing does not occur. Strong secondary structures such as hairpins might prevent the probe from functioning properly. This analysis can be done today with commercial programs such as Oligo 6.6. In general intermolecular hybridisations have a low probability to affect the functionality of the probes.

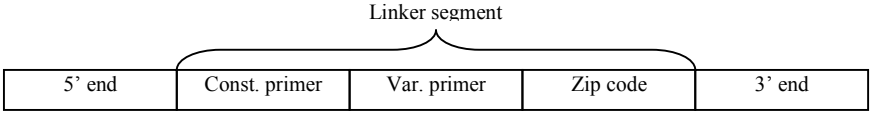

Figure 2. The different parts of a padlock probe. The 5’ and 3’ends are complementary to the target sequence and the nucleotide complementary to the variable position in the target is at the extreme position of the 3’ end. The constant primer is used for amplification purposes, the variable primer is allele specific and the zip code is locus specific.

This way of designing probes requires much work. A lot of time has to be invested in the process of designing each single probe. When it comes to using multiplexed padlocks on a microarray a considerable amount of time must be given only to the design of the probes. To speed up the process and to facilitate the spread of the padlock technique it is important to automate the process of designing padlock probes.

Methods

Algorithms

Smith-Waterman

To detect complementary regions within the probe the Smith-Waterman algorithm is used (9). It is adjusted to search for complementary regions instead of regions of similarity. The Smith-Waterman algorithm is a modified version of the Needleman-Wunch algorithm and uses dynamic programming to find an optimal local alignment. A local alignment aligns the pair of regions within the sequences that are the best complements to each other, given the choice of scoring matrix and gap penalties. In this case the two sequences are the complete probe sequence and its reverse.

Smith-Waterman is mathematically rigorous. It is guaranteed to find the best scoring alignment between the pair of sequences being compared. It does this by constructing a two dimensional table of partial alignment scores. The tables have one dimension or axis for each sequence. Each cell in the table contains the score for the best partial alignment that ends with the pair of sequence residues (one from each sequence) that correspond to that cell in the table. That best scoring partial alignment will be extended to subsequent cells in the table only when it is the prior cell that results in the best scoring partial alignment for the subsequent cell. In this way all possible alignments are considered until they are proven inferior to a competing alignment that also involves aligning at least one of the same pairs of sequence residues. The final alignment is thus the best scoring alignment possible.

The Smith-Waterman is easily described in a recursive, mathematical equation

$$SW_{i,j} = \max\begin{Bmatrix} SW_{i-l,j-l} + s(a_i, b_j) \\ SW_{i-k,j} + g_j \\ SW_{i,j-k} + g_i \\ 0 \end{Bmatrix}$$

$SW_{i,j}$ is the Smith-Waterman score for the partial alignment ending at residue i of the first sequence, a and residue j of the second sequence, b. In computing the Smith-Waterman score the four terms in the equation must be calculated and the term with the maximum value selected. This gives the highest possible score at that point. The first term, $SW_{i-1,j-1} + s(a_i, b_j)$, corresponds to extending the alignment by one residue from each sequence. The second term, $SW_{i-k,j} + g_j$, corresponds to extending the alignment by including residue j from sequence b and inserting a gap of k residues in length, aligned to end with residue j of sequence b, into sequence a. The third term, $SW_{i,j-k} + g_i$, is the equivalent term for inserting a gap into sequence b. The fourth term, zero, is added because the partial scores within the table are not allowed to become negative.

The Smith-Waterman algorithm places no restriction on the alignment other than that it has a positive score in terms of the scoring table used to score the alignment.

"Scoring matrices" for nucleic acids

A key element in evaluating the quality of a pairwise sequence alignment is the "scoring matrix", which assigns a score for aligning any possible pair of residues. The result of an alignment is thus dependent on the scores given different matches and mismatches. There is a vast number of matrices that have been tailored to detect similarities among sequences that diverged by different degree. Commonly used scores for nucleic acid sequences are the PAM 47 scores and PAM 50 scores (10). These however are used when scoring for optimal similarity, as in BLAST. I have not been able to find a suitable matrix for scoring regions of complementarity between two nucleotide sequences, but I have empirically found a set of scores that seems to work fairly well on available data (table 1.)

The design of my matrix is based on how many hydrogen bonds are formed between the different pairs of nucleotides. This assumption only gives the relative sizes between different matches and not mismatches. Negative numbers are given to mismatches so that at least two matches are required to extend the alignment past a mismatch. Since the sequences that are aligned are short, and to be able to detect short but strong alignments I have scaled the matrix to rise above background noise.

	A	C	G	T
A	-8	-8	2	4
C	-8	-8	6	-8
G	2	6	-8	-5
T	4	-8	-5	-8

Table 1. Default scoring matrix in the makepad program.

In the case of RNA alignment, scores for U are assumed to be similar to DNA scores for T.

Scoring Bulges

In most alignment and search programs, the gap penalty consists of two terms, the cost to open the gap and the cost to extend the gap. The selection of appropriate scores for bulges in the sequences, the gap penalties, is as important as selecting the scoring matrix for the algorithm to be biologically relevant. A detailed statistical theory for gapped alignments has not been developed, and the best gap costs to use with a given scoring matrix are determined empirically (11).

In this application one intuitively understands that the alignments are rather short. Bulges most likely destabilise the hybridisation of one part of the probe with another, so I want to force alignments to have relatively few gaps. To achieve this and to minimise the number of bulges that are longer than one nucleotide, I have chosen a linear model giving the same penalty for creating or opening a gap as for extending a gap. The default gap penalty is 17.

Melting temperature calculation

The nearest neighbour model

The NN model for nucleic acids assumes that the stability of a given base pair depends on the identity and orientation of neighbouring base pairs. The helix-coil transition works as a zipper; after an initial attachment, the hybridisation propagates laterally. Two duplexes with the same base pairs could have different stabilities and conversely two duplexes with different sequences but identical sets of Crick's pairs will have the same thermodynamics properties (12). This model also assumes a two state model, duplex and random coil.

For oligonucleotide heteroduplexes 16 different NN parameters describe all possible matches between opposite strands. Two different initiation parameters are added to account for initiation of duplex formation and other sequence independent effects (including differences between terminal and internal NN:s and counterion condensation). One for duplexes with terminal A·T and another for duplexes with terminal G·C (13). (An additional entropic penalty for the maintenance of the C2 symmetry of self-complementary duplexes is also included.) NN enthalpy and entropy parameters for the effects of dangling ends were recently published (14).

To account for mismatches between the two strands another 4^4 - 16 = 240 NN parameters are required. Unfortunately this data set is not yet complete (15, 16, 17, 18). Notably the thermodynamic data set corresponding to tandem mismatches (i.e. immediately adjacent mismatches) is still incomplete. The influence of salt and loop effects is also still under investigation (19).

Prediction of the melting temperature

T_m is defined as the temperature at which half of the strands are in the double helical state and half are in the random coil state. The T_m of an incoming nucleic acid duplex is computed from the predicted enthalpy and entropy using nearest neighbour thermodynamics. The module first computes the hybridisation enthalpy and entropy from the elementary parameters of each Crick's pair

$$\Delta H = \delta H_{initiation} + \sum \delta H_{Crick'spair}$$

$$\Delta S = \delta S_{initiation} + \sum \delta S_{Crick'spair}$$

ΔH and ΔS are the enthalpy and entropy for helix formation, respectively. Then the melting temperature is calculated using the formula

$$T_m = \frac{\Delta H}{\Delta S + R \times \ln(C_T)} - 273.15$$

R is the molar gas constant (1.987 cal / grad C *mol), C_T is the total oligonucleotide strand concentration. For non-self-complementary molecules C_T is replaced by $C_T/4$, if the strands are in equal concentration. 273.15 is subtracted to get the temperature in degrees Celsius.

According to SantaLucia, the salt correction is found to be sequence independent but to be dependent on nucleotide length. The salt correction changes the entropic term without modification of enthalpy (20).

$$\Delta S = \Delta S_{([Na^+]=1M)} + 0.368 \times (N-1) \times \ln[Na^+]$$

Where $\Delta S_{([Na+]=1M)}$ is the original ΔS and N is the length of the duplex. The mismatching pairs are also taken into account. The thermodynamic parameters are however only available for single internal mismatches. Whenever double mismatches or mismatches in the first or last positions are encountered an error message will appear and the program will transform the mismatch to a match to be able to calculate the melting temperature.

Approximate T_m calculation

When too many parameters are missing an approximate T_m calculation is performed. This is done by using the %(G+C) model. The number of G and C nucleotides in the nucleic acid strand are counted and the T_m calculated according to Wetmur (21).

$$T_m = 81.5 + 16.6 \times \log \frac{[Na^+]}{1 + 0.7 \times [Na^+]} + 0.41 \times \%(G+C) - \frac{500}{Length}$$

The formula is slightly different for RNA/RNA T_m calculation.

$$T_m = 78 + 16.6 \times \log \frac{[Na^+]}{1 + 0.7 \times [Na^+]} + 0.7 \times \%(G+C) - \frac{500}{Length}$$

Note that this kind of T_m calculation is increasingly incorrect when the length of the duplex decreases. Moreover, it does not take into account nucleic acid concentration, which is a severe mistake.

The application

The program is written in C++. This language was chosen since it will produce a fast program and since four students had started out working on this program as a project work during the autumn of 2000 (22). Their work was fundamental for the structure of the program. Makepad was compiled with the GNU C++ compiler (version egcs-2.91.66) and run under LINUX (Red Hat 6.2) on an Intel Pentium III processor. Parts of the code is written using the standard template library (STL), notably the vector and string classes have been used. It is crucial that the compiler has access to the standard template library (STL). Should the STL not be present the code will have to be rewritten.

The program follows a one-way path (figure 3). At first a parameter file is read, containing all data necessary to initiate an object of the central class Probe_pair. When the parameters are read they are checked and a probe is created. The length of the 5' and 3' arms are adjusted according to what was specified in the parameter file, but not lower than the ligation temperature, also specified in the parameters file (see Appendices 1 and 4). Before choosing a zip code to insert in the linker segment adjacent to the 3' end, the program checks for a history file. If such a file is present, it is copied to a temporary history file to keep track of which zip codes not to insert. At this point the program enters a loop and a zip code is chosen from a file with zip codes. The probe now holds all necessary data and is tested for intramolecular complementarities in the Smith-Waterman module. The most stable alignment is put into the T_m calculation module and the melting temperature calculated. If the set of sequences in the probe give strong intramolecular complementarities the program will loop back and test new zip codes until an acceptable one is found.

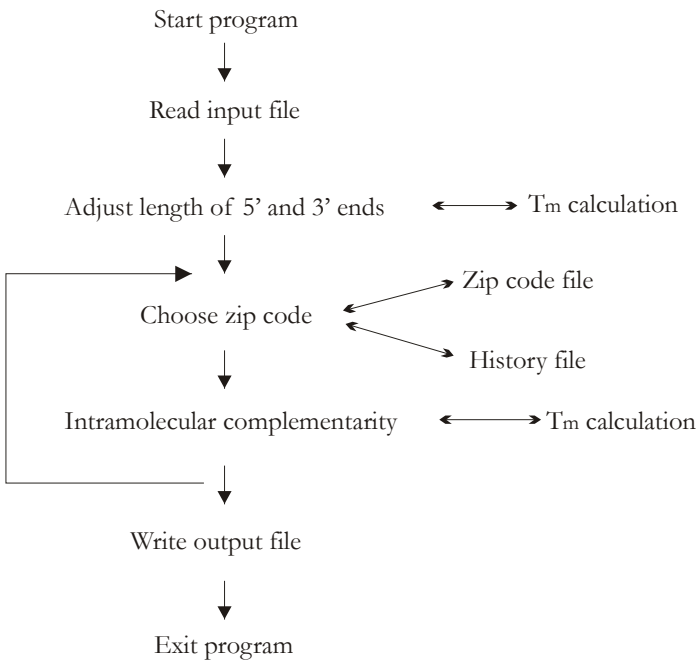

Figure 3. Structure of the makepad program

Directory structure

Each set of probes that should be designed together is stored in the same directory. This set of probes is called a project. To assure that different zip codes are used for the different probes in a project they should all have the same history file specified in their individual parameter files. The history file stores all zip codes that have been used in a project to make sure they are not used again. If it has been specified in the parameter file, the output file will also be placed here.

Four directories contain common files supplied with the program. The directories are the **settings** directory, which contains the nearest-neighbour parameter files and the zip code files, the **doc** directory with documentation, the **src** directory with source files and the **bin** directory with binary executables (figure 4).

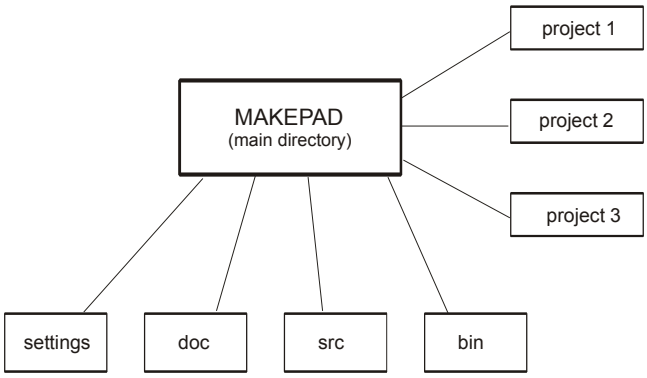

Figure 4. Directory structure of MAKEPAD

Error handling

The Probe_pair class keeps lists of comments to each probe. Vectors of strings are used to keep warnings and error messages that are connected to each probe. These messages are printed as part of the output of each probe. The error handling is similar in all modules.

The Probe_pair class

To make the program able to create two probes for one target sequence, with two different versions of the nucleotide at the snp position, a pair of probes is the central class and not just one probe. The two probes will thus have the same zip code but the primer in the middle of the linker sequence will be different as well as the last nucleotide in the 3' end, corresponding to the other allelic variant. In order to construct a pair of probes, the parameter file must contain three extra parameters. These are "OTHER_BASE_AT_SNP_POSITION", "SECOND_PROBE_NAME" and "PRIMER_2"(see Appendix 4).

The probe is built up of sequences. The probe arms, the zip codes and other components are objects of the class Sequence. In the Probe_pair class there are functions to handle the order of the individual sequences and to turn the set of sequences into a single long sequence. The class also contains an object of the Parameters class. This class is used to get input data from a parameter file. Finally the Probe_pair class has a number of variables to store information on the alignments from the Smith_Waterman module.

The Sequence class

All sequences are stored as objects of the Sequence class. The class has variables for sequence name, length, kind of nucleic acid and the string of nucleotides. It also contains a variety of functions to manipulate the variables of the class, to create complimentary or reversed copies of the sequence and to determine if part of sequences are complementary.

The Parameters class

The Parameters class is mainly a class to read and store all variables that are read from the parameter file (see Appendices 1 and 4).

FileIO

There are essentially three files that are used to store and retrieve information in the program. *History file* refers to a file which may be present when the program starts, or will be generated during the run. It contains the zip codes that have been used to design previous probes. *Zip code file* is the file in which all zip codes are kept. The third file is a temporary file, *temporary history file*. It serves the same purpose as *history file*, but will hold all zip codes that have been tested, and not only the ones that have been used. The *temporary history file* is removed when the program ends. It is crucial that these files are written in the right format. The format of the *zip code file* is a tab-separated list:

```
ZIP Name ZIP        sequence, 5' to 3'    Intensity  Comment
ProbeSet00005       GAGTAGCCTTCCCGAGCATT  High
ProbeSet00007       AAACCATCGACTCACGGGAT  High
ProbeSet00008       ATTGACCAAACTGCGGTGCG  High
```

Both the *history file* and the *temporary history file* use the same syntax, although the generated *history file* has a bit more information than the *temporary history file*. For each zip code entry there is a number of lines. The first line is a > followed by the name of the zip code. The second is the sequence of the zip code (FASTA format):

```
>ProbeSet00010

AACAACGATGAGACCGGGCT

>ProbeSet00020

ACTCCAGTGCCAAGTACGAT

>ProbeSet00022

GGCTCACGTCTTATTTGGGC
```

The *history file* also contains the complete sequences of probes that were designed using this zip code.

```
>ProbeSet00005

GAGTAGCCTTCCCGAGCATT
1 GGGATTATAAAGAACTGTTGCTCGACCGTTAGCAGCATGAttCCGAGATGTACCGCTATCGTGAGTAGCCT
TCCCGAGCATTTCTTCTGGGCTAATTACAGC
2 GGGATTATAAAGAACTGTTGCTCGACCGTTAGCAGCATGAttAGAGCGCATGAATCCGTAGTGAGTAGCCT
TCCCGAGCATTTCTTCTGGGCTAATTACAGA
```

FileIO is a collection of functions and constants used in the program. As the program starts it will look for the *history file* specified in the parameter file. If the file exists it will copy the *history file* to a *temporary history file* using the *copy_file()* function. The function, *choose_sequence()* and a couple of sub functions are used to choose an unused zip code from the *zip code file*. Chosen sequences are appended to the *temporary history file* as they are tested. When the program succeeds to find a good set of sequences, the used zip code is appended to the *history file* and the *temporary history file* is deleted with the function *remove_file()*.

Melting

The melting temperature module was inspired by the program Melting, written in C by Nicolas Le Novère (23). Melting was condensed and rewritten in C++ by Christoffer Hamilton (22).

The original program uses nearest neighbour (NN) thermodynamics to calculate the T_m (see Methods, Algorithms, The nearest neighbour model) of a sequence and its complement, both given as input arguments. If this cannot be done, it will switch to an approximate T_m calculation using the %(GC) method. The program must be able to report the T_m of any pair of nucleotide strings, even pairs with gaps, dangling ends and mismatches. There are mainly four problems that must be considered in the calculations.

- Many mismatch NN parameters are still not determined
- There is no functional model to allow gaps in the sequence
- The increase in stability due to loop structures is not taken into account
- The effect of mismatches in the extreme ends of hybridised regions is unpredictable

As long as calculations are performed on completely complementary sequences the module will report correct T_m values. This is also true for sequences with single internal mismatches and dangling ends. To deal with gaps the function *remove_gaps()* was written. If a gap is encountered it will be removed from the sequences. Later, when the actual calculation is done, the algorithm will punish positions where gaps were removed by dividing the values of the NN parameters by a factor 2 for each position. This will slightly decrease the stability of the hybridisation, reflecting the presence of a bulge.

Parameters for the effects of dangling ends, unmatched terminal nucleotides, have recently been published and the program now deals with these structures. Other mismatches however still cause problems. External mismatches, i.e. mismatches on the two extreme end nucleotides, are unpredictable and all cases are possible (12). For instance, the duplex

```
A           T                                  
 GTGAGCTCAT      is more stable than      AGTGAGCTCATT
 TACTCGAGTG                               TTACTCGAGTGA
T           A
```

External mismatches are simply ignored in this version of makepad. In case of external mismatches the program will give an error message and calculate the T_m on the rest of the sequence.

The lack of double internal mismatch parameters reduces the sensitivity of the calculations. Whenever a missing parameter is required, the module will transform a mismatched nucleotide to a match and give a warning to circumvent the lack of the parameter. This will increase the stability of the hybridisation, reducing the sensitivity of the algorithm. The parameter max_transformations determines how many transformations are allowed before the module switches to approximate T_m calculation mode. max_transformations is currently set to 4, but should perhaps depend on sequence length.

Finally loop structures are not considered in this version, since enthalpy and entropy parameters for different loop sizes have not been found. Most likely the stability of a hairpin is dependant on some constant, the hairpin stem and loop lengths and the salt concentration.

Smith-Waterman

The search for complementary regions within the probe is done using the Smith-Waterman algorithm (see Methods, Algorithms, Smith-Waterman). The main function in this module is *do_smith_waterman()*. This function takes the Probe_pair, a scoring matrix, the position of the zip code and the gap penalty as input arguments. On entry into the function, it creates a full-length sequence and complement of the individual probe sequences. Using the scoring matrix to score matches and mismatches between individual nucleotides, an integer matrix with score values and a char matrix with the traces are created and calculated. The function *trace_back()* is then used to find the highest scoring trace in the calculated scores matrix. The trace is returned to the main function and *check_if_overlap()* check if the same nucleotide is involved twice in the alignment. It is physically impossible for a nucleotide to hybridise to itself or to be involved in two alignments, so these traces have to be sorted out. It is also necessary for loops to be a couple of nucleotides long, since a couple of nucleotides are required to bend the nucleic acid. Minimum loop size is one nucleotide for DNA and two nucleotides for RNA (24).

Once the strongest non-overlapping trace is found, corresponding to the most stable hybridisation within the probe, the T_m is calculated for this trace. Using the function *locate_aligned_position()* in the Probe_pair class the program determines what parts of the probe is involved in the alignment. Knowing what parts of the probe cause the strongest hairpin and the T_m of this structure the probe is either accepted or rejected. If T_m is less than LIGATION_TEMPERATURE – TEMP_SPAN (both parameters specified in parameter file, see Appendices 1 and 4) the probe is accepted and a 0 returned, no matter where the alignment is located. If T_m is less than LIGATION_TEMPERATURE – 10 and the alignment neither includes the zip code nor the 3' end, the probe is also accepted and a 1 is returned. If the probe is rejected, -1 is returned to the main program and a new zip code will be tested.

If a pair of allele-specific probes is to be constructed the same procedure is performed for both probes and they are only accepted if both probes are accepted with the same zip code.

Integer and char matrices

In the alignment search both integer and char matrices are used. A char matrix is used to store the traces and an integer matrix is used for the calculated score values. The two classes are equivalent, the only difference is the variable type. The parameters of the classes are an array of arrays of integers/chars, a matrix of integers/chars and the size of the matrix. An integer/char matrix is created while given a size and then the values are set and fetched using the operator.

The scoring matrix

Another matrix structure is the scoring matrix. This matrix holds the values that are given to matches and mismatches between nucleotides in the Smith-Waterman module. Default values are set in the constructor, but other values can be read from file (see Appendix 4, Command line arguments). The syntax of the file is rather simple. Each line should contain one pair of nucleotides, separated by a slash (see Appendix 2). Immediately after the last nucleotide, write a colon and the value you want to assign this particular match or mismatch.

```
A/T:4
A/C:-8
```

etc.

Results and discussion

Test of experimentally verified probe designs

A set of eight pairs of probes designed by hand and used in several experiments have been put through the program (table 2). As expected only weak intramolecular complementarities are detected for the majority of the probes. Three probes however seem to have relatively stable hairpins, 3a, 4a and 4b. A closer look at these alignments reveals that all three contain a gap, external mismatches and several internal mismatches. These values are therefore most likely erroneous. The T_m of 4a and 4b were calculated using the approximate T_m mode and since the alignments were rather long with a high GC content the melting temperature is high.

The influence of bulges on the hybridisation is most likely stronger than what is reflected in the present model. Once a better model is developed to calculate the melting temperature of nucleotide strings with gaps it should be incorporated to improve the sensitivity of the algorithm. The same is true for the transformation of mismatched nucleotides. It is a way to circumvent the lack of mismatch parameters, on the expense of sensitivity. This problem will be less important and finally vanish when more parameters are published and incorporated. The approximate %(GC) mode is increasingly incorrect for mismatched and for short sequences, normally giving mismatched sequences higher T_m than the NN model. This problem will also decrease as the amount of mismatch parameters is increased.

The one important physical parameter not taken into account is the loop size. The stability of a hairpin is greatly affected by the size of the loop and incorporating a model to account for this increase in stability will improve the liability of the output.

As a negative control a probe was designed with complementary 5' and 3' ends. When attempts were made to design this probe, the program would loop through all zip codes but fails to design a good probe.

Probe name	T_m strongest alignment	T_m 5'end	T_m 3'end
1a	1*	59.52	65.55
1b	1*	59.52	64.11
2a	28.16	64.60	70.13
2b	21.02	64.60	69.42
3a	40.92	73.81	74.82
3b	12.91	73.81	74.58
4a	52.32**	65.64	63.21
4b	52.32**	65.64	64.19
5a	17.31	69.51	76.00
5b	29.12	69.51	72.97
6a	32.66	78.96	77.09
6b	1*	78.96	75.33
7a	23.17	75.47	78.55
7b	17.57	75.47	77.38
8a	32.41	68.38	72.55
8b	32.75	68.38	71.62

Table 2. Results from a test of eight pairs of hand designed padlock probes.
* Negative results are reported as 1.
** Calculated using %(GC) mode.

Designs of probes against new target sequences

A new set of seven probe pairs was designed. To each target sequence twenty different designs were generated. Only four of these sets have so far been evaluated. The designed probes were tested for secondary structures in a reference program, Oligo 6.6. Some of the output probes have strong secondary structures according to the reference program, but others look very promising (Johan Banér, personal communication). The conclusion is that this program needs further refinement. The results however indicate that the current version of makepad can be used as a first screening among the list of zip codes. Makepad can create a list of candidate probes to be tested in a reference program. This provides a good way of screening for candidate designs and reducing the amount of time required to design a probe.

Future developments

The default scoring matrix and the gap penalties in the Smith-Waterman module are rather blunt and could undoubtedly be improved and refined. The construction of a matrix only on the basis of formation of hydrogen bonds is of course less precise than using for instance forces created between the matched/mismatched nucleotides. Using other physical parameters such as Gibbs's free energy, the enthalpy or the entropy as a base for the design of the matrix might produce a more accurate scoring matrix.

The system of reading from and writing to different files makes the program slow when a couple of hundred zip codes have been tested. The speed is decreasing due to the fact that each chosen zip code is compared to the list of tested zip codes in the temporary history file. As this list gets larger, the amount of operations required to choose a new zip code increases. Keeping the complete list of zip codes in a STL vector, list or similar data structure would make it possible to speed up the program by sorting the individual zip codes. Such a data structure would however be rather large.

The current version of makepad detects what parts of the probe are involved in the strongest local alignment given by the Smith-Waterman algorithm. However the program makes little use of this information. It is clear that some parts of the probe can hold a more stable secondary structure than others and yet still be functional. It may even be desirable to design a probe with a certain secondary structure in a specific part of the probe. When more nearest-neighbour thermodynamic parameters are available and the melting temperature calculations more reliable, it would be interesting to implement functions that limit secondary structures in some parts of the probe and try to impose weak secondary structures on other parts.

The human genome sometimes contains more than one copy of a gene and sometimes copies of smaller sequences such as the ones targeted by the padlock probes exist in multiple sites. Therefore it might be a good idea to align the target sequences against the human genome to search for homologies. It ought not matter if parts other than those complementary to the target of the probe bind to the genome, since the probes are in excess and since circularisation of the probe only occurs when the 5' and 3' ends are brought into juxtaposition. Thus unspecifically bound probes will not disturb the reaction.

Users that are new to the Linux operative system might find it a bit difficult to use the program. To facilitate the spread and use of the program it might be a good idea to develop a user-friendly graphical interface. Another feature to meet these needs would be an online service where users could submit a form and get results back by e-mail.

The program as it looks today is designed for the specific needs of the users. As the needs change and as extensions are desired, the program will have to change. The design of the program was chosen to facilitate incorporations of future extensions and changes. It is my belief that the modular structure of the program and the easily changeable main function will satisfy future programmers that are to implement these changes.

References

1. Lander ES, Linton LM, Birren B, Nusbaum C, Zody MC, Baldwin J, Devon K, Dewar K, Doyle M, FitzHugh W, Funke R, Gage D, Harris K, Heaford A, Howland J, Kann L, Lehoczky J, LeVine R, McEwan P, McKernan K, Meldrim J, Mesirov JP, Miranda C, Morris W, Naylor J, Raymond C, Rosetti M, Santos R, Sheridan A, Sougnez C, Stange-Thomann N, Stojanovic N, Subramanian A, Wyman D, Rogers J, Sulston J, Ainscough R, Beck S, Bentley D, Burton J, Clee C, Carter N, Coulson A, Deadman R, Deloukas P, Dunham A, Dunham I, Durbin R, French L, Grafham D, Gregory S, Hubbard T, Humphray S, Hunt A, Jones M, Lloyd C, McMurray A, Matthews L, Mercer S, Milne S, Mullikin JC, Mungall A, Plumb R, Ross M, Shownkeen R, Sims S, Waterston RH, Wilson RK, Hillier LW, McPherson JD, Marra MA, Mardis ER, Fulton LA, Chinwalla AT, Pepin KH, Gish WR, Chissoe SL, Wendl MC, Delehaunty KD, Miner TL, Delehaunty A, Kramer JB, Cook LL, Fulton RS, Johnson DL, Minx PJ, Clifton SW, Hawkins T, Branscomb E, Predki P, Richardson P, Wenning S, Slezak T, Doggett N, Cheng JF, Olsen A, Lucas S, Elkin C, Uberbacher E, Frazier M et al: **Initial sequencing and analysis of the human genome.** *Nature* 2001, **409** (6822), 860-921.
2. Antson D-O: **Genotyping RNA and DNA Using Padlock Probes.** Acta Universitatis Upsaliensis. Comprehensive Summeries of Uppsala Dissertations from the Faculty of Medicine 1050. 40pp Uppsala ISBN 91-554-5057-1, 2001.
3. Nilsson M, Malmgren H, Samiotaki M, Kwiatowski M, Chowdary B P and Landegren U: **Padlock Probes: Circularizing Oligonucleotides for Localized DNA Detection.** *Science* 1994, **265**(5181), 2085-2088
4. Landegren U, Kaiser R, Sanders J and Hood L: **A ligase-mediated gene detection technique.** *Science* 1988, **241**(4869), 1077-1080
5. Landegren U and Nilsson M: **Locked on target: Strategies for Future Gene Diagnostics** *Ann Med* 1997, **29**, 585-590.
6. Banér J, Nilsson M, Mendel-Hartvig M and Landegren U: **Signal amplification of padlock probes by rolling circle replication** *Nucleic Acids Res* 1998 **26**: 5073-5078
7. Antson D-O, Isaksson A, Landegren U and Nilsson M: **PCR-generated padlock probes detect single nucleotide variation in genomic DNA** *Nucleic Acids Res* 2000, **28**, e58
8. Affymetrix, GenFlex^{TM} Tag Array, Technical note No. 1
9. Smith T.F. and Waterman M.S: **Identification of common molecular subsequences.** *J. Mol. Biol.*1981, **147**, 195-197
10. Hugh B. Nicholas Jr et al. **Sequence Analysis Tutorials: A Tutorial on Searching Sequence Databases and Sequence Scoring Methods.** http://www.psc.edu/biomed/training/tutorials/sequence/db/index.html, 02 April 2001
11. Web page at NCBI, http://ncbi.nlm.nih.gov/BLAST/matrix_info.html, March 2001
12. Sugimoto N, Katoh M, Nakano S, Ohmichi T, Sasaki M: **RNA/DNA hybrid duplexes with identical nearest-neighbor base-pairs have identical stability.** *FEBS Letters* 1994 **354**, 74-78
13. Allawi H.T, SantaLucia J: **Thermodynamics and NMR of internal G·T mismatches in DNA.** *Biochemistry* 1997 **36**, 10581-10594

14. Bommarito S, Peyret N, SantaLucia J: **Thermodynamic parameters for DNA sequences with dangling ends**. *Nucleic Acids Res* 2000 **28**, 1929-1934

15. Allawi H.T, SantaLucia J: **Nearest Neighbour Thermodynamic Parameters for Internal G·A Mismatches in DNA.** *Biochemistry* 1998 **37**, 2170-2179

16. Allawi H.T, SantaLucia J: **Thermodynamics of internal C·T mismatches in DNA.** *Nuc. Acid Res.* 1998, **26**(11), 2694-2701

17. Allawi H.T, SantaLucia J: **Nearest Neighbour Thermodynamics of Internal A·C Mismatches in DNA: Sequence Dependence and pH effects.** *Biochemistry* 1998, **37**, 9435-9444

18. Allawi H.T, SantaLucia J: **NMR solution structure of a dodecamer containing single G·T mismatches.** *Nuc. Acid Res.* 1998, **26**(21), 4925-4934

19. Homepage of SantaLucia J: http://sun2.science.wayne.edu/~jslsun2/, April 2001

20. SantaLucia J: **A unified view of polymer, dumbbell, and oligonucleotide DNA nearest-neighbour thermodynamics** *Proc. Natl. Acad. Sci.* 1998, **95**, 1460-1465

21. Wetmur J G: **DNA Probes: Applications of the Principles of Nucleic Acid Hybridisation.** *Criti. Rev. In Biochem. and Mol. Biol.*1991, **26**, 227-259

22. Hamilton C, Hälltorp G, Kindlund E and Osifo O: **Computer-aided Construction of Padlock Probes** 2001, Uppsala universitet, Inst. För informationsteknologi, Internrapport nr. 2001:2

23. Melting homepage, http://www.pasteur.fr/recherche/unites/neubiomol/meltinghome.html, Januari 2001

24. Gesteland R.F, Cach T.R and Atkins J.F: **The RNA World.** Second edition, CSHL Press, 1999.

25. Banér J, Nilsson M, Isaksson A, Mendel-Hartvig M, Antson D-O, Landegren U: **More keys to padlock probes: mechanisms for high-throughput nuclear acid analysis.** *Current Opinion in Biotechnology* 2001, **12**: 11-15.

Appendices

Appendix 1: Example of a parameter file

```
# An example file
PROBE_NAME:Example_1
SECOND_PROBE_NAME:Example_2
TARGET_SEQUENCE:gcaacagttctttataatcccGctgtaattagcccagaagaa
SNP:22
OTHER_BASE_AT_SNP_POSITION:T
MIN_ARM_LENGTH_FIVE_PRIME:20
MIN_ARM_LENGTH_THREE_PRIME:20
SODIUM_CONCENTRATION:0.1
PROBE_CONCENTRATION:0.0001
LIGATION_TEMPERATURE:50
TEMP_SPAN:20
#MIN_BIND_NUCLEOTIDES:4
# This is a comment
ZIPCODE_FILENAME:./settings/zip_codes.txt
ZIPCODE_ADDITIONAL_INFO_FILENAME:./Dir_one/example_history.txt
# 0 = DNA, 1 = RNA
DNA_OR_RNA:0
OUTPUT_FILE:./Dir_one/example_results.txt
#
CONSTANT_PRIMER:CTCGACCGTTAGCAGCATGA
PRIMER_1:ttCCGAGATGTACCGCTATCGT
PRIMER_2:ttAGAGCGCATGAATCCGTAGT
(END)
```

Appendix 2: Example of a scoring matrix file

```
A/A:-8
A/C:-8
A/G:2
A/T:4
A/U:4
C/A:-8
C/C:-8
C/G:6
C/T:-8
C/U:-8
G/A:2
G/C:6
G/G:-8
G/T:-8
G/U:-8
T/A:4
T/C:-8
T/G:-8
T/T:-8
T/U:-8
U/A:4
U/C:-8
U/G:-8
U/T:-8
U/U:-8
```

Appendix 3. Example of an output file

```
**********************************
Output of first probe
          Name    Example_1
          5' end  GGGATTATAAAGAACTGTTG
 Constant primer  CTCGACCGTTAGCAGCATGA
     Primer one   ttCCGAGATGTACCGCTATCGT
 ProbeSet00005    GAGTAGCCTTCCCGAGCATT
          3' end  TCTTCTGGGCTAATTACAGC

Complete sequence:
GGGATTATAAAGAACTGTTGCTCGACCGTTAGCAGCATGAttCCGAGATGTACCGCTATCGTGAGTAG
CCTTCCCGAGCATTTCTTCTGGGCTAATTACAGC
Tm in 5' arm: 59.51
Tm in 3' arm: 65.54
Melting temp in strongest alignment is: 14.10
Bases 90 to 100 align to bases 70 to 60
GGCTAATTAC
CCGATGAGTG

Comments to first probe: None

**********************************
Output of second probe
          Name    Example_2
          5' end  GGGATTATAAAGAACTGTTG
Constant primer   CTCGACCGTTAGCAGCATGA
     Primer two   ttAGAGCGCATGAATCCGTAGT
 ProbeSet00005    GAGTAGCCTTCCCGAGCATT
          3' end  TCTTCTGGGCTAATTACAGA

Complete sequence:
GGGATTATAAAGAACTGTTGCTCGACCGTTAGCAGCATGAttAGAGCGCATGAATCCGTAGTGAGTA
GCCTTCCCGAGCATTTCTTCTGGGCTAATTACAGA
Tm in 5' end: 59.51
Tm in 3' end: 64.11
Melting temp in strongest alignment is: 14.10
Bases 90 to 100 align to bases 70 to 60
GGCTAATTAC
CCGATGAGTG

Comments to second probe: None

**********************************
Comments to alignment:
Two copies of the first probe will bind to each other. Tm for this structure is 58.51
Two copies of the second probe will bind to each other. Tm for this structure is 57.55
```

Appendix 4. Running the program

This appendix is intended to serve as a manual for users of the program. It is written to be comprehensive for users with limited knowledge of how the Linux operative system works.

The two crucial items for the user to control are the parameter file and the arguments that can be given on the command line. To run the program, make sure you have a correct parameter file, let's call it param, in a directory of your choice, here called dir_one. Being in the main directory of the makepad program, simply type

./makepad dir_one/param

to launch the program.

Command line arguments.

Additional arguments can be given to change the default settings of the program. These arguments are given on the command line, immediately after the path of the parameter file. The arguments are

-s followed by the path to a file containing a scoring matrix that one wants to use (see Methods, The Application, scoring matrix).

-g followed by a number changes the gap penalty. Default is 17 (see Methods, Algorithms, Smith-Waterman)

-n specifies how many probes to construct

Example:

Let's say you want to design a probe based on the parameters in your parameter file param in the project directory dir_one. However you believe another scoring matrix, scores, is optimal for the design of this probe and you want to use a gap penalty of 15. Assume you want the program to design 10 probes that meet your criteria. If the file with the parameters for the scoring matrix scores is in the main directory of makepad then just type

./makepad dir_one/param –s scores –g 15 –n 10

Parameter file

The parameter file (Appendix 1) contains all information the user has to supply to design the probe. These parameters are stored in the class Parameters. Each object Probe_pair contains one object of this Parameter class. Some of the parameters in the file are compulsory, others are optional. The syntax of the file is simple. Each line contains the name and value of one parameter. First write the name of the parameter, then the value, separated by a colon:

PARAMETER:VALUE

The parameters can be written in any order. Comments are also allowed in the file. These are written starting with a #:

#This is a comment

The last line should be the single word (END). Eleven parameters are mandatory. If these are excluded from the file or if they are incorrect the program will quit with a warning. Following is a list of all parameters of the parameter file. It starts with the compulsory ones, followed by optional parameters. In the end are the parameters that are required if a pair of probes should be constructed.

TARGET_SEQUENCE

This is the 5' -> 3' target sequence that the probe should detect. Parts of the sequence downstream and upstream of the SNP position are used to construct the 5' and 3' ends, so the SNP position must be somewhere in the middle of the sequence. Both capital and lower case a, c, g, t and u letters are accepted.

SNP

Tells what nucleotide in the target sequence is at the SNP position. The first nucleotide in the target sequence is number 1.

DNA_OR_RNA

Specifies the kind of nucleic acid. O for DNA, 1 for RNA.

LIGATION_TEMPERATURE

Temperature at which the ligation reaction will be performed.

TEMP_SPAN

Tells how may degrees below the ligation temperature the T_m of the probe must be to be accepted by the program.

SODIUM_CONCENTRATION

Is the ion concentration used in the buffers. This concentration is used to calculate the T_m. It must be between 0 and 10 M. The effect of ions on thermodynamic stability of nucleic acid duplexes is complex and the corresponding functions are at best rough approximations. They are generally more reliable for $[Na^+]$ belonging to [0.1, 1]M. Note that the divalent ions, notably Mg^{2+} have a drastic effect and the calculations are better in the absence of such ions. Some authors showed that a mix of 0.15M NaCl with 10mM $MgCl_2$ could be equivalent to 1M NaCl. An artificial increase of the sodium concentration could be a way to proceed.

PROBE_CONCENTRATION

Concentration of the probe. Also necessary to calculate the T_m. Must be [0, 0.1]M.

MIN_ARM_LENGTH_THREE_PRIME

Determines the length of the 3' end. Probably a minimum of 6 bases is required for the ligase to be able to perform the ligation.

MIN_ARM_LENGTH__FIVE_PRIME

Determines the length of the 5' end.

ZIPCODE_FILENAME

Holds the path to the file where all zip codes are listed.

ZIPCODE_ADDITIONAL_INFO_FILENAME

Path to where a history file is located or where it will be put when the program generates it.

The following parameters are optional, but are recommended to be included in the parameter file.

PROBE_NAME

The name of the probe.

CONSTANT_PRIMER

The part of the linker sequence immediately adjacent to the 5' end. If this sequence is excluded from the parameter file the program will automatically put twenty T:s in this part of the linker sequence.

PRIMER_1

The middle part of the linker sequence. As for the constant primer, twenty T:s will be put here if the parameter is not included in the parameter file.

OUTPUT_FILE

Path to where an output file should be created. If this file already exist the new results will be appended to the end of the previous file. If this parameter is excluded the program will only print the results on the screen.

The last three parameters are required to construct a pair of probes with the same zip code. The two probes have the same target sequence, apart from the single nucleotide at the SNP position.

SECOND_PROBE_NAME

The name of the second probe.

PRIMER_2

As for primer_1 in the first probe, this is the middle part of the linker segment. This, apart from the SNP nucleotide is the only sequence difference between the two probes in the probe pair.

OTHER_BASE_AT_SNP_POSITION

Tells what allelic variant to detect with the second probe.

The program will accept probes with T_m of highest scoring alignment less than LIGATION_TEMPERATURE - TEMP_SPAN and probes where the zip code and 3'end are not involved in the alignment and T_m is less than LIGATION_TEMPERATURE – 10. Lowering these parameters will thus discriminate in favour of low T_m probes.