MicroFamily

for LINUX

Version 1 (15 July 2006)

Emese Meglécz
EGEE, CASE 36, Université de Provence, 3 place Victor Hugo, F-1331, Marseille, France
meglecz@up.univ-mrs.fr
http://www.up.univ-mrs.fr/Local/egee/dir/meglecz/MicroFamily.html

Overview

Microsatellite flanking regions are not necessarily unique sequences, but they may group into sequence families. This phenomenon seems to be widespread in Lepidoptera and also occurs in many other insect species (Meglécz *et al*. 2006). These microsatellites are likely to give multiple banding patterns during PCR amplifications, which can be very difficult to interpret. Therefore, identifying sequences that cluster together prior to primer design can save considerable time and money.
MicroFamily is a program designed for identifying flanking region similarities between different microsatellite sequences obtained from screening partial genomic libraries (Meglécz, 2006).
As a preparation for sequence comparison, sequences are edited by (i) replacing all characters other than ACGT by N (ii) by deleting the extremities if they contained more than two Ns in the ten most extreme base pairs, and (iii) by removing vector contamination if the sequence produced a BLAST hit against the UniVec vector base of NCBI (ftp://ftp.ncbi.nih.gov/pub/UniVec/). This latter step was not designed for precise removal of all vectors, but to avoid artificial similarities caused by vector contamination. The E-value (parameter that describes the number of hits one can "expect" to see just by chance when searching a database of a particular size) for the BLAST against the vector base can be specified by the user.
Microsatellites with motifs of 1-6 bp are identified. The minimal number of repeats for single base pair motifs and for all other motifs can be defined by the user.

The flanking regions of clean microsatellite-containing sequences are then compared by an all-against-all BLASTn analysis. The E-value can be specified by the user. Sequences are sorted into four categories based on the results of the BLASTn. They are classified as (i) *Unique* if no similarities were observed to any other sequences of the same dataset, (ii) *Redundant*, if the identity to another sequence was higher than 95% along the whole flanking sequence or (iii) *UnBLASTable* if the sequence had no hits at all (*i.e*. no significant similarity to any sequences), not even with itself. This is the case if the flanking region is too short or semi-repetitive (resembles to a microsatellite but there are not enough uninterrupted repeats). All non-redundant sequences that produced a significant hit with at least one different sequence were classified as (iv) *Grouped*.

Installation of the program

The MicroFamily is written in Perl and runs under Linux. It is a collection of small programs/modules, which use the following freely available programs:
Perl (ActivePerl 5.8.7.815;
http://www.activestate.com/Products/Download/Download.plex?id=ActivePerl)
BLASTN-2.2.13 (ftp://ftp.ncbi.nih.gov/blast/executables/ or
http://www.ncbi.nih.gov/BLAST/download.shtml)
CLUSTALw1.83 (Higgins *et al*. 1991;ftp://ftp.ebi.ac.uk/pub/software/unix/clustalw/ or

http://www.cf.ac.uk/biosi/research/biosoft/Downloads/clustalw.html)

Start by downloading and installing the above mentioned three programs. If you have a slightly older or newer version already on you computer, it will probably not cause any problem. The pathway to CLUSTAL and BLAST executables cannot contain a space.

Once the above three programs are installed, you can simply create a folder and unzip MicroFamily.tar.gz into it.
You should obtain the following files and folders

MicroFamily (folder)
- **datatemp** (empty folder)
- **temp** (empty folder)
- **test_files** (folder)
  - **sample_outfiles** (folder with 16 sample outfiles coming from 2 different run of MicroFamily)
  - 4 test fasta files

- class.pl
- clean.pl
- formatdb.log
- MicroFamily.pl
- readmeLINUX.rtf
- script_file
- script_file_standard
- set_file
- set_file_standard
- sub_ms_group.pm
- univec.fas
- univec.fas.nhr
- univec.fas.nin
- univec.fas.nsd
- univec.fas.nsi
- univec.fas.nsq

You are ready.
(Tip: keep the zipped file as well; the program is not protected from overwriting. If you make a bad move, you can just throw it away and unzip the file again)

Using MicroFamily

The program is designed for checking flanking region similarities among different microsatellite containing sequences. It works well for relatively small datasets typically produced by a screen of partial genomic library (few hundred sequences, of up to 1000 bp each). However, it is not designed for genome scale analyses.

Input format

All sequences that should be compared to each other should be put into a single fasta file.
The definition line of each sequence in the fasta file should start by '>' followed by the name/code of the sequence (without any spaces). Sequence codes can contain any numbers, letters or underscore, but all other characters are replaced by underscore(s) automatically. A space should separate all further information concerning the sequence and it will be ignored by the program.
Each sequence should follow its definition line. It can contain spaces and end of line brakes; these will be eliminated by the program.

Examples
>ACA025 325bp
ACTGATCTGATGTGATcgtagctagTGtgagtt
>P_mnemosyne02 reverse only
ATGTCGGATCGATCGCGCGCGCGCATGCTAGCTAGCTAG
CGATGCTAGCTAGCTAGCTA TCGATGCTAGCTATTCGGATACG
CTGATCGATA
>256
cgatgctagctagt cgtagctaggctag gctgatcgatcgatcg tcgatcgctagcta
ctgatcgatcgatc ctgatcggctaggct gctaggctaggctag tcgatctcgat

To run MicroFamily:
1. Copy your sequence file(s) in Fasta format into MicroFamily/datatemp
   - One fasta file should contain all the sequences you intend to compare;
   - You might place several fasta files into MicroFamily/datatemp/ but each of them will be analyzed separately;
   - Do not put any other files into MicroFamily/datatemp/
2. Make sure that MicroFamily/temp/ does not contain any *.VOW file or other files that can be overwritten (previous outfiles)
3. Open a Terminal and change the directory to MicroFamily
4. Type : 'perl MicroFamily.pl' then press enter

During the run:

The program will suggest a list of settings:
(Text in Bold is messages that appear on the screen)

A: Minimum number of repeats for 2-3-4-5-6-bp motifs:	**4**
B: Minimum number of repeats for single-bp motifs:	**8**
C: e value for vector search:	**1E-3**
D: e value for BLASTing sequences against each other:	**1E-40**
E: lowest identity for redundant sequences:	**0.95**
F: pathway to BLAST:	**/usr/local/bin/blast-2.2.13/bin/**
G: pathway to CLUSTAL:	**/usr/local/bin/clustalw1.83/**

Press Enter if all of the above settings are correct, or the appropriate capital letter if you want to change the settings

If you wish to change any of these settings type the appropriate capital letter. The program will ask you to type the new setting. Once you agree with all of the proposed parameters, press enter to start the program.

Aand B: Microsatellites will be identified only if they have a minimum number of uninterrupted repeats. You can specify this minimum number for all motifs that are longer than a single base pair (A) and for single bp motifs (B)
C: Sequences are blasted against the UniVec vector base of NCBI (ftp://ftp.ncbi.nih.gov/pub/UniVec/). You can change the E value and check the detailed results in MicroFamily/temp/*.vec file (see bellow) in order to choose the E value that is most convenient for you
D: E value for the all-against-all sequence comparison by BLASTn
E: If the sequence identity between the flanking regions of two sequences is higher than the value specified here, the shorter sequence will be considered as redundant
F: Gives the pathway to BLAST executables from the root
The default value for the pathway to BLAST is **/usr/local/bin/blast-2.2.13/bin/**
G: Gives the pathway to CLUSTAL executables from the root
The default pathway to CLUSTAL is **/usr/local/bin/clustalw1.83/**
These pathways are very probably different on your computer and you should specify the right pathways by typing F or G.

The program will memorize the settings you have specified during the last run, so once you have correctly specified the pathways, you do not have to re-type it all over again. However, if you have completely lost track of the correct formats by mistake, you can delete the lines found in set_file and replace them by the lines found in set_file_standard. In this case the program will suggest again the original default values.

The program checks if only numbers, letters or underscore are present in the sequence codes. All other characters are automatically replaced by an underscore and a message appears on the screen:

'/' is replaced by '_' in 18/178

Then it checks if none of the sequence codes are substrings of other codes (e.g. A1 is a substring of A10). This might cause errors during the run. If it is the case, it changes all numbers in the code (if shorter than three digits) into a three digit form (e.g. A11B2 into A011B002). If the problem of substrings still persists, the program will completely rename your sequences with an X letter Y digit number code, where numbers range between 1 and the total number of sequences in the file (e.g. PMN001-PMN124). In this latter case, the program will prompt you to enter a short string of letters (e.g. PMN) that will be the beginning of the sequence code, and also will be used later while naming the output files (e.g. MicroFamily/temp/PMN.info).

In this case, the following message will be displayed:
The codes of the sequences are incompatible with the program.
Sequences shall be renamed by a X letter Y digit code (e.g. PMN002).
Please, enter a short string (only letters) that will be the beginning of each of the codes:

Here you should type a string of letters of your choice (I suggest using few, but informative characters.

for example: PMN stands for ***Pmn****emosyne*).

For example, type:
PMN

The following messages will indicate the progress of the run and the name of the relevant outfiles (see bellow for further explanations).

temp/PMN.code: Correspondence between old and new codes
temp/PMN.fas: Fasta file with renamed but otherwise unchanged sequences
temp/PMN.vec: The results of a blast against the Univec vector base
temp/PMN.info: Information on the cleaned sequences
temp/PMN.clean: Microsatellite containing clean sequences in fasta format
temp/PMN.WOV: Microsatellite containing clean sequences in fasta format; Microsatellites are masked by Ns

***.code**and ***.fas** files are only produced if sequences have been renamed. Otherwise the name of your original fasta file - with the appropriate extensions - is used for naming outfiles.

Note: Occasionally the following type of message can appear:
temp/PCO.vec: The results of a blast against the Univec vector base
Overlap between Microsat in PCO28 and Vector >SAULB PCO
Vector:>SAULB PCO, positions:210-229, MS:183-408, OVERLAP!!!

This is very likely a false hit during the BLASTn against the UniVec vectorbase, where the vector matches a sequence either between two microsatellites or the matching region itself includes a microsatellite. In the example above, the adaptor SAULB PCO resembles to the sequence PCO28 at positions 210-229 (of the sequence). In the same sequence, microsatellites are detected between positions 183-408 of the same sequence. This hit is ignored in the cleaning process, but you should check manually if there is really no vector contamination in the given sequence.

During the cleaning of the original files (files in MicroFamily/datatemp), one MicroFamily/temp/*.VOW file is produced for each of the input files. ***.VOW** files contain only cleaned sequences with masked microsatellites and they are used as input files for the next step, to group sequences into four categories: Unique, UnBLASTable, Grouped, Redundant (see explications at the output files section and Meglécz *et al.* 2004, 2006)

The following message appears to indicate the progress of the analyses:

Grouping sequences. This step might take a few minutes. Please, wait!
(The time needed to accomplish this step increases with (i) the number of sequences analyzed and (ii) the proportion of the grouped sequences)

Grouping sequences of temp/PCO.WOV
temp/PCO.txt: Results of the all-against-all BLASTn
temp/PCO.ide: Pairwise similarities between grouped sequences
temp/PCO.clas: Categories for each sequences

Note:
A possible error message can be: "**The system cannot find the path specified**". This means that you did not give the pathways to BLAST and/or CLUSTAL correctly. If this message appears both before and after the line "**Grouping sequences. This step might take a few minutes. Please, wait!",** it means that the pathway to BLAST is incorrect. If it appears only after it, the pathway to BLAST is correct but not the one for CLUSTAL.

Output files:

All of the output files are found in the MicroFamily/temp/ folder

temp/*.clean

Fasta file with only the microsatellite containing cleaned sequences

All non-ATGC characters are changed into N
If the ten bps at each of the ends contain more than 2 Ns, bases are deleted till the last N
Potential vector contaminations are cut off

temp/*.WOV

Same file as ***.clean** but Microsatellites are marked by Ns. (It is the input file for grouping sequences)

temp/*.vec

The results of the BLAST against the UniVec base of NCBI.
This step was designed to avoid false similarities among flanking regions due to vector contamination, and not for precise removal of all external sequences. By using far lower E values for the all-against-all BLAST than for the Vector search (e.g. the default values of E = 1E-3 for vector search and E = 1E-40 for all-against-all analyses) we can be sure that flanking region similarities are not the result of vector contamination, but there is no guarantee that short stretches of vectors are not remaining at the extremities of the sequences (especially if it contains sequencing errors). Alternatively, due to false hits, some of the real flanking regions might be removed. By decreasing the E-value for the vector search, the probability of finding matches decreases, thus your sequences can remain contaminated. By increasing the E-value the probability is higher for getting a false hit, thus eliminating part of the true flanking regions.
You can also add your adapter/linker sequences to the UniVec base(MicroFamily/Univec.fas). In this way adapters/linkers can be automatically removed from the sequences. However, if your linker/adapter sequences are short (less then 20 bp) and contain sequencing errors, they might not be picked out by the BLASTn.

temp/*.info

Information on the cleaning of each sequence.
This file can be easily imported in Excel, as columns are separated by semicolons.
The following information is given:

CODE: Sequence code
Orig_Length: Original length of the sequence

NON_ATGC: Characters other than ATCGN that were present in the sequence
Ns_BEG: The number of base pairs deleted from the beginning of the sequence because it had a high proportion of Ns (more than 2/10)
Ns_END: The number of base pairs deleted from the end of the sequence because it had a high proportion of Ns (more than 2/10)
Vector_pos_BEG: The position of last bp cut off from the beginning of the sequence because of potential vector contamination.
Vector_ref_BEG: Number referring to the vector (vector names are given at the end of the file)
Vector_pos_END: The position of first bp cut of from the end of the sequence because of potential vector contamination. (10000 indicate that no vector was identified at the end of the sequence)
Vector_ref_END: Number referring to the vector (vector names are given at the end of the file)
New_Length: Length of the sequence after cleaning
MICROSAT MOTIF and POS: each microsatellite motif and positions (in the cleaned sequence), separated by commas (*i.e*. AC 14-27, AC 34-55, CA 56-71, CA 74-81, CA 115-122,)

temp/*.code

This file is provided only if sequences have been renamed by the program.
This file can be easily imported in Excel, as columns are separated by semicolons.
First column: new sequence codes
Second column: original sequence codes

temp/*.fas

This file is provided only if sequences have been renamed by the program.
Renamed, but otherwise unchanged (not cleaned) sequences in fasta format.

temp/*.txt

Output file of the all-against-all BLASTn sequence comparison. You can check the length and position, of the similar fragments, E-value, strand orientation etc. between each pair of sequences.

temp/*.ide

This file can be easily imported in Excel, as columns are separated by semicolons.
This file gives pairwise identities between grouped sequences (Microsatellites are masked, only flanking regions are considered).

temp/*.clas

This file can be easily imported in Excel, as columns are separated by semicolons.

Sequences codes are given in the first column.
The results of the sequence classification (unique, UnBLASTable, grouped, redundant) either with or without the identification of redundant sequences are found in the following three columns.

As a first approach, all sequences are classified into one of the following three categories (**Redundants are not identified**):
Unique: No similarities were observed to any other sequences of the same dataset
*UnBLASTable:*if sequence had no hits at all, not even with itself. (If the flanking region is too short or repetitive, but not a perfect microsatellite, BLAST masks the region and ignores it.)

*Grouped: S*equences that produced a significant hit with at least one different sequence.

RAW_results: (4thcolumn)
Results of the first grouping for each sequences (Unique, UnBLASTable or grouped). For grouped sequences all sequence names that produced a reciprocal hit are given.

GROUPS: (2ndcolumn)
Same as RAW_results but instead of giving hits for each grouped sequences, they are sorted into groups. Groups are defined in the largest sense, thus if sequence *A* is similar to sequence *B* and *B* is similar to *C*, *A* and *C* are grouped together even in *A* and *C* do not produce a reciprocal hit. This is only an approximation, that generally works well for smaller datasets with relatively few grouped sequences, but quite meaningless for large ones.

The 3rd column **(with_REDUNDANTS)** gives the result of the classification **with the identification of redundant sequences**.
Sequences that are very similar to each other along the whole flanking region might be different alleles of the same locus or the same allele sequenced more then once. Therefore it is justified to distinguish them from grouped sequences that are different loci. Sequences are classed as *Redundants* if the identity to another sequence was higher than 95% (default value) along the whole flanking sequence. However, this is an arbitrary limit (for discussion see Meglécz *et al*. 2004), since flanking regions of different alleles of the same locus can be more or less conserved. Thus you should always keep in mind that a sequence classified as redundant might be genuinely redundant sequence due to picking up the same locus several times in the screening, but it can also be a recently duplicated locus that will produce a multiple banding pattern.
For redundant sequences the code of a highly similar sequence is given (e.g. redundant_AJ809365). For grouped sequences all sequences that produced a hit is given after removing redundant sequences. If in a group there were only two sequences and one of them was classified as redundant the other was re-classed as unique.

A few guidelines to choose sequences for primer design

Although the use of grouped microsatellites as genetic markers is not impossible (Hoekstra *et al.* 1997), they are unlikely to be good candidates for single locus genotyping purposes since flanking region similarities can lead to multiple locus amplifications and unclear banding patterns. Thus I think best candidates for primer design are sequences that were classed as unique both with and without the identification of redundant sequences (with_REDUNDANTS (3rd column) and RAW_results (4th column) in temp/*.clas file). If you do not have enough usable sequences, the second best option is to select sequences that were classed unique after the identification of redundant sequences (with_REDUNDANTS (3rd column) in temp/*.clas file), but they were grouped without the identification of redundants. They might be genuinely unique loci for which multiple copies have been sequenced or different alleles of the same locus. However, they can also be recently duplicated loci, that give multiple banding patterns.

If you are desperate, you can also choose grouped sequences, but try to choose the ones where only few sequences are found in the same group (GROUPS (2nd column) in temp/*.clas file). In this case, check the position of similar region among sequences in the temp/*.txt file.

Alternatively, you can try to take advantage of flanking region similarities by designing one primer that matches more than one locus coupling with other primers, each of which are unique to one single sequence. This is a cost and time effective alternative for sample analyses, but might need a considerable amount of time to establish a reliable protocol.

References

Higgins DG, Bleasby AJ, Fuchs R (1991) CLUSTAL V: improved software for multiple sequence alignment. *Cabios*, **8**, 189-191.

Hoekstra R, CriadoFornelio A, Fakkeldij J, Bergman J, Roos MH (1997) Microsatellites of the parasitic nematode Haemonchus contortus: polymorphism and linkage with a direct repeat. *Molecular & Biochemical Parasitology*, **89**, 97-107.

Meglécz E, Péténian F, Danchin E *et al.* (2004) High similarity between flanking regions of different microsatellites detected within each of two species of Lepidoptera: *Parnassius apollo* and *Euphydryas aurinia*. *Molecular Ecology*, **13**, 1693-1700.

Meglécz, E. 2006. MICROFAMILY: A computer program for detecting flanking region similarities among different microsatellite loci. *Molecular Ecology Notes* (06/251) *In Press*

Meglécz, E., Anderson, A., Bourguet, D., Butcher, R., Caldas, A., Casel-Lundhagen, A., Cœur d'Acier, A., Dawson, A.D., Faure, N., Fauvelot, C., Franck, P., Harper, G., Keyghobadi, N., Kluetsch, C., Muthulakshmi, M., Nagaraju, J., Patt, A., Péténian, F., Silvain JF., Wilcock, HR. 2006. Microsatellite Flanking Region Similarities among Different Loci within Insect Species. *Submitted*