CopyMap Manual v1.0

Author: Sebastian Zöllner; szoellne@umich.edu

Basic overview

CopyMap is a hidden Markov algorithm to infer copy number variation from a sample of hybridization intensities. The program combines the information across individuals by estimating a transition matrix between every pair of markers using a Baum-Welch algorithm. The algorithm consists of two steps, the first step estimates the underlying parameters, the second step estimates carrier status in each individual based on the estimated parameters.

The algorithm used here is described in
Henrichsen CN[*]*, Vinckenbosch N*[*]*, Zöllner S*[*]*, Chaignat E, Pradervand S, Ruedi M, Kaessmann H, Reymond.(2009) Segmental copy number variation shapes tissue transcriptomes. Nat Genet 41: 424-429*
Please cite this paper if you consider the program useful.

Presently the program is still in beta testing, thus both the program and this manual are a little rough. Thus, if you see parts of the program or this manual that could be improved upon, please let me know.

Compiling the program

You should have downloaded a C-code with the name hmm*x*.c, where *x* is the version number. To compile the program type:

```
gcc ./hmmx.c -lm -O3 -o CopyMap
```

This will create an executable named copymap in the same directory. In the following we assume that **CopyMap** is the name of the executable.

Technical Details

Statistical Model

The program considers the hybridization signal in each individual as one realization of a hidden Markov model (HMM). The HMM has three hidden states, baseline (0), insertion (1) and deletion (2). The hybridization signal for each state is a realization of the same distribution with means (a_0, a_1, a_2) dependent of the hidden state; the program allows for different signal distributions of this intensity. Other than classical HMMs, we do not assume an identical transmission matrix between every pair of probes. Rather we estimate a specific transition matrix between every pair of probes using the Baum-Welch algorithm (Baum LE, 1970). Transition probability between 0 and 1 or 2 can be interpreted as the population frequency of an insertion/deletion.

To reduce the number of false positives, the program can require a minimum length (option – m) for each CNV and assign a prior probability (option – P) of each location carrying a CNV.

To infer the carriers and the locations of CNVs, the program offers two methods. Method 1 is the classical Viterbi algorithm (Viterbi, 1967)(option -V), which will use the location specific transition matrix to analyze each sample separately, estimating carrier status and begin and end of each CNV independent of all other samples. Note that this method still used the location-specific transition matrix and thus still combines evidence across individuals. However, it allows CNV boundaries to vary between individuals. Method 2 is a consensus method. It scans the location specific transition matrix for regions with a high probability of transitioning into a CNV (threshold set by -t option) and calls these regions to have variable copy number. Then the signal intensity of each individual is used to assign a probabilistic carrier status.

Input file format:

All data used to run CopyMap is entered in a single file. In this input file, every line starting with a hash (#) will be ignored by the program and can be used for comments. The first line of the input file starts with a **P** (for parameters) followed by the number of individuals and the number of probes for each individual, the numbers separated by a space. The input file may then contain any of the following lines:

- Starting with **M** showing the label used for missing data
- Starting with **N** listing the names of all individuals
- Starting with **L** listing the location of all probes.

Including lines L and N will make the output more readable.

Each following line consists of the hybridization intensities for one individual in order of the probes. If any line does not contain the specified number of probes, the program will report an error and abort.

```
#short toy example
P 5 3
# 5 probes 3 individuals
L 12138 12390 12505 12945 13880
# Location of the 5 probes. This
#line is optional.
M 9999
#flag for missing data. This line
#is optional
N mouse1 mouse2 mouse3
#Names of the three individuals
#listed below
-0.271 0.431 -0.801 0.093 0.242
0.175 -0.118   9999 0.067 0.135
0.009 -0.370 -0.376 -0.098 0.243
```

Running the program

CopyMap uses the command line to enter parameters and options. To run the program type

```
./CopyMap –option1 –option2 –option3.
```

A typical command would be

```
./CopyMap -r20 -s30 -p18000 -T4 -m5 –P0.005 –iin_chr.txt
```

This would analyze a dataset of 30 individuals (-s), each individual with 18000 probes (-p) taken from an input file “in_chr.txt” (-i). The program would model hybridization intensity as a mixture of two normal distributions (-T4), setting a minimum length for each CNV of 5 (-m) and a prior of variable copy number of 0.005 (-P). The program would run 20 rounds of the Baum-Welch forward-backward algorithm (-r). A list of all options is given in the table below.

Runtime estimates

The runtime of the program depends on five parameters, the number of EM steps, the sample size, the number of probes, the minimum length of each CNV and the minimum distance between CNVs, and it scales roughly linearly with each of those parameters. The table below shows a couple of typical runtimes on a single 2.33 MHz processor with 16 Mb of memory.

EM steps	Sample size	probes	Minimum length	Runtime
30	100	10,000	5	9 min
30	100	10,000	10	20 min
30	500	10,000	5	40 min
30	100	50,000	5	40 min

Command line options

Generally command line options come in two flavors, flags (F) and input parameters (P). Flags indicate that the program should run in a certain way, for example model genotyping error. Parameter values are put in directly after the line option, **without** a space in-between. More detailed description of some input parameters is given below.

Input parameters:

s	P	Sample Size
p	P	Number of probes for each individual
N	P	Number of repeat measurements for each probe
T	P	Type (distribution) of hybridization signal
M	P	Code for missing data in the input file
i	P	Name of the input file

Note that the s and p parameters duplicate the P-line in the input file. These numbers have to be identical; otherwise the program will exit with an error message.

Parameters for the hidden Markov model Baum-Welch:

r	P	Rounds of EM; 30 is generally sufficient to achieve convergence, but for quick analyses, shorter numbers can be chosen
a	P	average factor of the insertion/deletion state to X
m	P	Minimum number of probes covering each CNV
d	P	Minimum distance between two CNVs
F	F	Run second hmm to improve estimate of p-values
H	F	Run the program as a classical hmm with one transition matrix for all locations, uses the Viterbi algorithm to call.
B	F	No preset factor for the insertion/deletion state; if this is set, any value of a is ignored

Parameters for CNV-calling

t	P	sets the minimum frequency threshold at which a CNP is called at to X.
O	F	Flag to allow overlap between CNVs of the same type
C	F	Cleanup: Remove all singletons after analysis
V	F	Viterbi algorithm used to infer carrier status

General Parameters:

f	F	Fast analysis, uses more memory
o	P	output file --- name of the output file; default: hmmout.txt
h	F	help; a quick overview over these options

Parameter Choices

To identify CNVs without too many false positives, several parameters have to be set: Most important are the minimum length of each CNV and the mean signal of a CNV.

m The prime method for reducing the number of false inferences, a minimum length of each CNV can be set using the –**m** option. The algorithm forces all CNV to cover at least X probes where X is the number set behind m. Computation time increases with m, so m>>20 is not advisable. Appropriate values of m depend on the technology used to generate the intensity data. The algorithm will not automatically miss all CNVs that are shorter than the minimum length; however the boundaries will be set wrong.

d Setting the minimum distance between two consecutive CNVs (**d** option) should make it less likely that big CNVs are cut in two. However, it also increases computation time.

P Prior probability of a CNV starting at a given locus. Setting higher values of P reduces the number of false positives. Below are some simulated false positive numbers for different values of m and P for s=100, p=10,000.

m	5	10	5	10	5	10	5	10
P	1e-2	1e-2	5e-3	5e-3	1e-3	1e-3	5e-4	5e-4
Viterbi FP	18	11	8	6	2	1	1	1
Joint FP	0	0	0	0	0	0	0	0

a Setting the mean signal intensities to

$$a_1=a_0+X\sigma \qquad a_2=a_0-X\sigma,$$

where σ is the standard deviation of the hybridization signal for estimating the transition matrix.

F If a parameter is set with -**a**, the program can be run with the -**F** flag to run a second Baum-Welch algorithm that keeps the transition matrix and estimates more

	appropriate values of a_1, a_2. Using this option improves the estimates of carrier status and the Viterbi algorithm.
f	If a parameter X is set with –a, the program can be sped up by setting the –**f** flag. This will increase the memory requirement but speed up the program.
T	Four models of signal distribution are selected by adding a number after the –**T** option. 1 binary 2 Normally distributed date 3 t-distributed date with 4 degrees of freedom 4 Distributed as a mixture of two normal distributions with identical means 5 Insert single point probabilities of copy number status directly. See below for more details on this option. If the empirical signal distributions for option 5 are not available, we have generally made the best experiences using the mixture of two normal distributions.
B	This Baum-Welch algorithm can be run to also estimate (a_0, a_1 a_2) (option **B**). However for most datasets, this will result in fitting to noise in the baseline distribution rather than in detecting actual CNVs.
t	For calling CNVs after the transmission matrix in the probabilistic algorithm, the program considers all pairs of markers with transition probability > X, set with –**t**X. This is roughly consistent with setting the minimum population frequency of an inferred CNV to X. Setting this parameter high can screen out a lot of true positives, especially when the a parameter is set high.
V	CNV status will also be estimated using the Viterbi algorithm. This will create a second output file containing calls based on these calculations.

Output

The program produces ongoing output on the standard output (usually the screen). It also summarizes all results into several output files. In the following we first explain the standard output and the individual files produced. We show snippets of output files generated by the same analysis as examples.

Standard output

The Standard output consists of 3 segments, the input options, the summaries of the Baum-Welch algorithm and a summary of inferred CNVs.
The first part is a printout of some of your input options.
Following that are summaries from every round of the Baum-Welch algorithm, starting with the line `Initializing HMM` and ending with `Forwards-Backwards algorithm`

`finnished`. The information presented her can usually be ignored. How to use this output to improve program performance is described below.

The next block titled `Heuristic CNV calling` gives an overview of the CNVs inferred with the calling heuristic. The table to the right shows the meaning of each summary statistic.

The last block titled `Viterbi summary` is only generated if the -V option is used. Each line represents one CNV. As the Viterbi-algorithm analyzes one individual at a time, each CNV is called in one individual at a time. The output lists the individual, the start and the length of the CNV. The last lines in this output are housekeeping messages reflecting an orderly termination of the program.

CNV	A running number of the detected indel
Type	Whether the CNV is a duplication (type 1) or a deletion (type 2).
start	first probe covered by the CNV
length	Total number of probes covered by the CNV
frequency	Probability of transitioning into the CNV

Main output file

The default name of this file is *hmmout.txt*; it can be changed using the -o option. The file starts with a paragraph listing input file, time and program version used. Then it displays the mean signal for each state and in the next line the standard deviation. Each line after that shows one inferred CNV. The first 5 elements of that line show basic statistics for the Indel (see table).

After this, each number in the line shows the carrier probability for one individual in the sample. The example below shows the output for one duplication assessed in a sample of 15 individuals. Individuals 1 and 4 have evidence for segregating the duplication (P=0.78 and P=0.85). Individual 10 has only little evidence (P=0.04) while individual 13 has strong evidence (P=1.0)

```
Indel 2 type 1 start probe 2796
length 10 frequency 0.087 Carrier
status: 0.78 0.00 0.00 0.85 0.00
0.00 0.00 0.00 0.00 0.04 0.00 0.00
1.00 0.00 0.00
```

Support file

```
CNV 2 type 1
0.78: 0.81 0.94 0.85 0.04 0.92 0.34 0.85 0.96 0.04 0.56 av:0.63
0.00: 0.66 0.63 0.34 0.50 0.17 0.76 0.15 0.08 0.28 0.21 av:0.38
```

The default name of this file is *support_hmmout.txt*; the second part of the name is set using the -o option. This large file can serve as a sanity check on generated results. It lists the signal intensities for each individual in CNV areas summarized in the main output file. CNVs are listed in the same order as in *hmmout.txt*. Each line in this file represents one individual. The first number in each line shows the probability of this individual carrying the CNV. After the colon, the program lists the intensities of all probes covered by this CNV and the last column shows the average intensity. The example here shows the two lines corresponding to the first two individuals in the figure above.

Viterbi output

The default name of this file is *vitt_hmmout.txt*; the second part of the name is set using the -o option. This file is only generated, if the -V option is selected. The first paragraph lists input file, time and program version used. The next line lists the order of all probes that have been inferred to be copy-number variable in at least one individual. Each of the following lines represents one individual, showing the copy number status estimated at the probe. the example shows parts of the position line and the copy number line for individual 1. similar to the output.txt file the individual is inferred to carry a CNV starting at probe 2796. However, other than in the overall analysis, the Viterbi-algorithm estimates the CNV to only cover 8 probes; probe 2804 is estimated to not be copy number variable.

```
 1238    1239    2796    2797    2798    2799    2800    2801    2802    2803    2804    2805    3695
0       0       1       1       1       1       1       1       1       1       0       0       0
```

Optimizing CopyMap

Then the steps of the hmm are shown. The important part here are the first two numbers, as they show the likelihood of the hmm (it is the first number *10^second number). This number should not change much for the last few steps of the hmm, otherwise r is too low. The next number is the proportion of the chain that is estimated to be not part of a CNV. If this ever drops below 0.98, you probably have P or m set too low.
The other numbers are not important.

This allows you for example to notice if a particular probe produces a lot of outlier values or if the presence or absence of a CNP is inferred because of a few probes or if all the probes show a signal.

Additional Options

```
#short toy example 2 for T5 option
P 3 3
L 12138 12390 12505
M 9999
#See above for the P L M options
N mouse1 mouse2 mouse3
#Names
0.90 0.03 0.07 0.80 0.20 0.00 0.88 0.03 0.09
0.83 0.17 0.00 0.55 0.44 0.01 0.88 0.08 0.04
0.44 0.02 0.54 0.22 0.00 0.78 0.31 0.01 0.68
```

Example 2: Input file of emission probabilities of a hmm with 3 states. Each shaded area contains the probabilities of states 1, 2, 3 for one probe.

In many cases, assuming a distribution for the underlying signal is questionable at best. Alternatively, it may be possible to use data with known CNV status to estimate the signal distributions under that status and use these empirical distributions in the hmm. This is done setting the parameter T5. Then the program will expect an input file as presented in Example 2, containing the probabilities for each state at that position. Note that the probabilities for all states at one probe add up to one. If they don’t’ the program exits with an error message. Not that you can only use this option if you do not have more than one signal per probe. The first probability for each probe is the probability of baseline status. As there is limited memory assigned to each input line, the program is not able to read in more than 3 digits per probability.