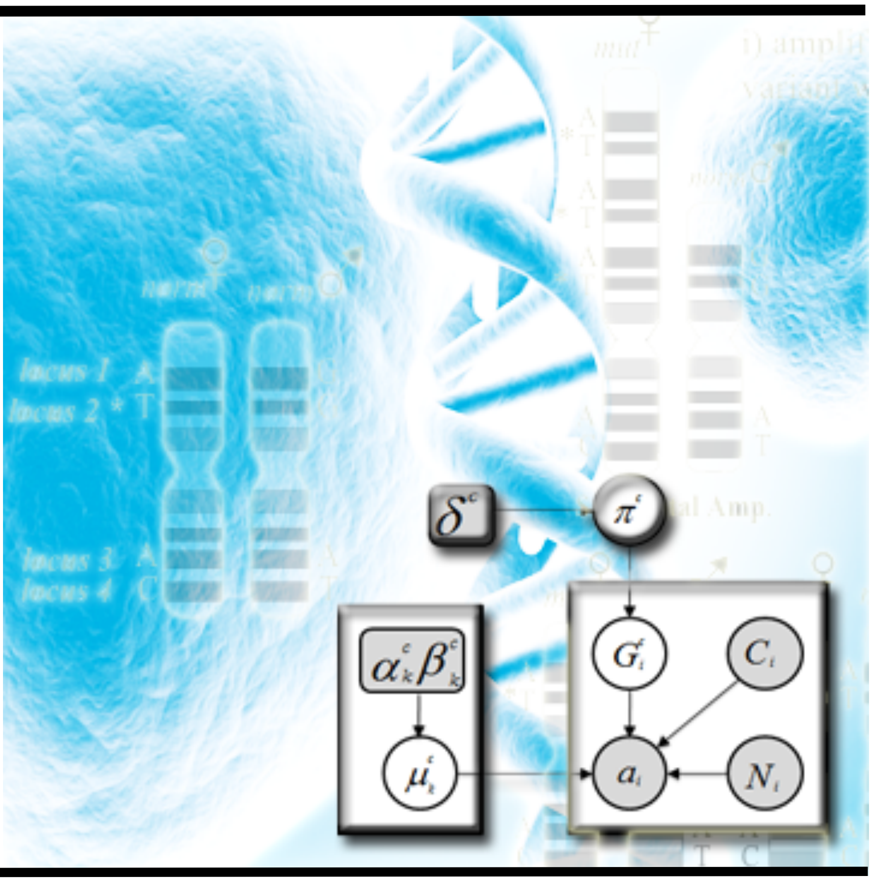
user guide

CoNAn-SNV: Copy Number Annotated SNV discovery

updated June (2010) by Anamaria Crisan



Introduction

CoNAn-SNV is a probabilistic framework for the discovery of single nucleotide variants in WGSS data. This software explicitly integrates information about copy number state of different genomic segments into the inference of single nucleotide variants. A full description of the model and its performance evaluation is available in the following manuscript:

Mutation discovery in regions of segmental cancer genome amplifications with CoNAn-SNV: a mixture model for next generation sequencing of tumors.
Anamaria Crisan[1], Rodrigo Goya[1,2], Gavin Ha[1], Leah M Prentice[1], Arusha Oloumi[1], Janine Senz[3], Thomas Zeng[2], Kane Tse[2], Allen Delaney[2], Marco A Marra[2], David G Huntsman[3], Martin Hirst[2], Sam Aparicio[1,*], Sohrab P Shah[1,3,*]

CoNAn-SNV can operate independently of CNA segmentation algorithm presented in the manuscript so long as it receives input in the format described in *Getting Started* section. We do however provide our segmentation algorithm on our website and it's documentation is present in the appendix of this manual. Should you choose to use an independent method, please carefully consult the *Segmentation Algorithm Notes* section to understand how this may affect the output. Finally, CoNAn-SNV is designed and tested on WGSS data and we do not recommend its use for WTSS data (RNA-seq).

Contact Information
To report any bugs, suggest improvements or for additional support please contact:

Sohrab P Shah, PhD
MSFHR Research Fellow
Centre for Translational and Applied Genomics
Molecular Oncology Breast Cancer Research Program
BC Cancer Agency, Vancouver BC Canada
+1 604 877 6000 x2589
sshah@bccrc.ca

Getting Started

CoNAn-SNV takes as input a pileup file either in a BAM or MAQ format and a file specifying CNA segments. In the CoNAn-SNV manuscript, Supplemental Table 2 provides an example of the segment input file. It outputs the positions, reference base, non-reference base and a probability for each of the genotypes existing in that CNA segments.

Pileup File Specification

When generating a pileup file it is necessary to specify the **–s** option in order to ensure that mapping and base qualities are included. The manuscript uses a framework similar to SNVMix1 in order to allow for a conceptual comparison of the different algorithms. The implemented version, however, is subsisted on the SNVMix2 underlying framework and will use mapping and base qualities to weight allelic counts. Additionally, we do not recommend using the **–c** option for the pileup creation because it creates additional columns that the CoNAn-SNV model does not handle.

Segmentation Algorithm Input Format

The input format for the segmentation algorithm requires a chromosome, start and end co-ordinates and finally a numeric encoding of the copy number state. For example:

Chromosome	Start Positions	End Positions	CNA State
1	114861349	247188490	5
5	170568420	180647645	4

The numeric encoding by our segmentation algorithm is as follows: 2 (NEUT/LOSS); 3(GAIN); 4(AMP); and 5 (HLAMP). This can vary depending on the segmentation algorithm used (see *Segmentation Algorithm Notes*).

CoNAn-SNV Parameters
The following explains the parameters for the CoNAn-SNV, which can also be accessed by using the –h parameter in the CoNAn-SNV in the command line.

Parameter	Explanation
-m	Model file that has the π and μ parameters for each CAN state (specified after parameter, i.e. mu 2 is parameter for NEUT/LOSS) mu 2 0.xxxxxxxx 0.xxxxxxxx 0.xxxxxxxx mu 3 0.xxxxxxxx 0.xxxxxxxx 0.xxxxxxxx 0.xxxxxxxx pi 2 0.xxxxxxxx 0.xxxxxxxx 0.xxxxxxxx pi 3 0.xxxxxxxx 0.xxxxxxxx 0.xxxxxxxx 0.xxxxxxxx For classification: *-m* parameter provides model parameters For training: *-m* parameter serves as output, storing the learned parameters
-i	Input file (i.e. the pileup file – see previous page for correct specifications). Does not have to be defined; pileup input can also be piped i.e. <STDIN>.
-c	Copy Number Segment File. See above for file specifications. Required in Training and Classification.
-S	Specifying the total number of copy number states. For example, the numerical encoding for a HLAMP in our model is 5 (we still consider LOSS 1 but analyze it as NEUT – or 2) so we would enter –S 5 for our segmentation algorithm
-o	Classification output file. If not specified, output will print to <STDOUT>
-T | -C	T (Training) and C (Classification). If not provided, the default is -C
-p	Input pileup format. Can take on 2 values m : MAQ pileup s : SAMTOOLS pileup
-t	Full described in SNVMix2 manuscript. For the CoNAn-SNV model implemented in manuscript, we use –t SNVMix1. In practice, we typically use –t MB. The options are listed below: mb Lowest between map and base quality m Filter on map, and use as surrogate for base quality b Filter on base quality, take map quality as 1 M Filter on map quality but use only base quality Mb Filter on map quality and use both map and base qualities MB Filter on map quality AND base quality SNVMix1 Do not consider contribution of m or b to allelic counts conan Analyse data with CoNAn-SNV mode
-q	Minimum required base quality (otherwise filtered out)
-Q	Minimum required mapping quality (otherwise filtered out)
-M	Training only. List of hyper parameters required for Training the model. An example is provided on the website.
-f	By default CoNAn-SNV will output only positions that contain at least one instance of the variant allele, even if that position does not have a SNV. Indicating *–f* will output all positions in a pileup.

Running CoNAn-SNV

It is possible to use CoNAn-SNV for classification as well as algorithm training. Suggested methods for Training the algorithm are available in the SNVMix and CoNAn-SNV papers as well as alternate strategies in various machine learning texts. In brief, training allows the probabilistic framework to better model the dynamic ranges of an individual tumor's landscape. There are a number of training strategies for such models as are frequently described in machine learning mathematical texts.

CoNAn-SNV is distributed as part of the SNVMixsuite package. This package is capable of running not only CoNAn-SNV but also SNVMix1 and SNVMix2.

⇒Training the model

./SNVMixsuite –S 5 –c *segment_file.dat* –i *training_pos_samtools.pileup* –m *output_modelfile.dat* **–t CoNAn** –q 20 –Q 10 –p s –M *hyperparameters.dat* -T

⇒Classifying with the model

./SNVmixsuite –S 5 –c *segment_file.dat* –i *samtools.pileup* –o *conan_snv_output.dat* **–t CoNAn** –q 20 –Q 10 –p s -C

In most instances, specifying **–f** is not necessary because positions containing no variant at all is generally not interesting. However, if you wish to assess performance then *–f* must be selected in order to ensure proper true positive and true negative sets. Others, true negatives with 0 variant alleles will be excluded

Post-processing considerations

The default final output from CoNAn-SNV is NOT an exhaustive list of SNVs, but a list of all co-ordinates containing more than one variant . A post-processing script may be desired to extract the desired variants. An example of such a script is available on the website. Here is an example of how it works.

```
chr:pos  ref  nref                 paa          pab            pbb        call
1:3204    C     A       C:20,A:1,0.9999955211,0.0000044789,0.0000000000,1
```

The above is a line of output from CoNAn-SNV. By summing the values of the variant containing states (p*ab* + p*bb*) we have a p(*SNV*). If p(SNV) > threshold, then we include the SNV in our analysis, otherwise we exclude it. The threshold can be set by any means. In our studies we set it such that the false positive rate on the training data was less than 1%.

Segmentation Algorithm Notes

It is possible to use different segmentation algorithms to provide CNA state information rather then relying on the method presented in the paper. As previously mentioned this will work so long as the input format is: chromosome, start and end position, and a numerical encoding of the CNA state. The numerical encoding of the CNA state can have a significant impact on the output data, and so it is important to understand several limitations prior to moving foreward with a different algorithm

1. Why not just use raw copy number? Why summarize segments with "annotations"?

Algorithms such as CoNAn-SNV are subject to challenges existing in model selection problems (which can be found described in greater detail in various machine learning texts). Using all possible raw copy numbers complicates the model selection problem because it introduces too many models and ultimately makes the algorithm uninformative. Consider our current schema: there is a genotype state space that represents high-level amplifications and a separate one for other CNA states. Using the same schema, but substituting the annotated copy number with the raw copy number, would create a genotype state-space for raw copy number, say, 15 and another separate one for raw copy number 16. The natural inclination is that there is no meaningful distinction between these raw copy numbers to require imposing a separate genotype state-space for each. Although, the true raw copy in the genome may be consistent, the data after sequencing and processing may not be. Further more raw copy number is variable from position to position, thus the dynamic range across the entire genome would be very large and difficult to accommodate.

Within a segment of, say, high-level amplification, positions will have variable depth due to issues with sequencing and alignment. An appropriate algorithm should accurately define segment boundaries, however it too can have problems of over-segmentation or conversely poor resolution.

2. How do I know I've selected a good CNA segmentation tool?

Consider the frequency of events that are occurring. In the lobular carcinoma presented in Shah et al. (2009): 30.2% of the genome was predicted as loss/neutral, 44.5% was gain, 19.1% amplification and 4.2% high-level amplification. In this genome, gains were so prevalent because it is triploid, but events like amplification and high-level amplification are still shown to be rare. If there were to be another level of amplification, for example "ultra high level amplification" it would likely encompass an even smaller proportion of the genome. The problem then becomes the model selection problem stated in the previous point. It is likely possible to have 6 or even 7 states in total; however beyond it is possible that the data may be over segmented and caution is advised.

3. Is it possible to use other technologies to inform copy number state?

It may be possible to use an orthogonal analysis tool, such as an array, to identify segmentation boundaries for CNA states for WGSS data. We note in our manuscript that oncoSNP identifies and classifies CNA segments similarly to our own NGS tool. We are doing experiments with this, but it is still in phases of preliminary investigation. It is important to identify that array platforms have their own biases and difficulties that must be overcome by their segmentation algorithms.