CVTree
(Standalone Version)
User's Manual

Guanghong Zuo and Bailin HAO

August 13, 2018

CVTree stands for **Composition Vector Tree** which is the implementation of an alignment-free algorithm to generate a dissimilarity matrix from comparatively large collection of DNA or Amino Acid sequences, preferably whole-genome data, for phylogenetic studies.

There are two versions of the program:

1. **CVTree Web Server** which has been published twice in the Web Server Issues of *Nucleic Acids Research*, [Qi *et al.*, 2004*a*] and [Xu and Hao, 2009]. The lastest CVTree Web Server, CVTree3 Web Server [Zuo and Hao, 2015], have two identical but independent installations:
   - Fudan University, Shanghai:
     `http://tlife.fudan.edu.cn/cvtree/`
   - Beijing Institute of Genomics, Beijing:
     `http://cvtree.net`
     `http://bigd.big.ac.cn/cvtree`
2. **CVTree Standalone Version** which is provided to those who are interested in the intermediate results, e.g., the collection of all CVs, or deal with extremely huge datasets of their own. We provide also a few options and scripts that were not available in the Web Server versions.

1 The Installation

1.1 Preparation

- cmake $\geq$ 2.8.4

- g++ $\geq$ 4.8 or other compiler supporting C++11 standard
- compiler with support openmp for parallel
- Library: libz, netcdf, netcdf_cpp

1.2 Compiling

Unzip or checkout the source files, Obtained the compiling option by cmake

- unzip the package file and change into it
- mkdir build and change into it
- cmake .. or add some options you wanted, e.g.:
  -DCMAKE_INSTALL_PREFIX=/usr/local
- make
- make manual
- make install

2 Programs and Command-Line Options

All the programs was implemented in C++.

2.1 cvtree: the all in one command

2.1.1 The Flow of cvtree command

The cvtree command is the main command for the software. It generate the the phylogeny tree from the fasta files directly.

2.1.2 The command usage

- cvtree – Generate nwk tree from input genomes

```
cvtree
[ -d <dm> ]          Output distance matrix format, defaut: <Method><Suffix
[ -t <nwk> ]          Output distance matrix format, defaut: <Method><Suffi
[ -G <dir> ]         Input genome file directory
[ -g faa ]           the type of genome file, defaut: faa
[ -V <cvdir> ]       Super directory of cv files
```

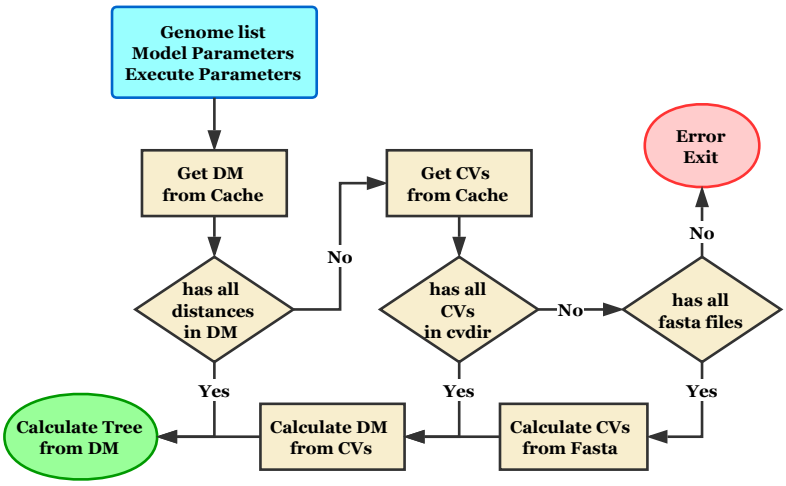

Figure 1: The Flowchart of the cvtree command

```
[ -i list ]          Genome list for distance matrix, defaut: list
[ -k '3 4 5 6 7' ]   values of k, defaut: N = 3 4 5 6 7
[ -r <matrix> ]      Reference distance matrixs, splite with ','
[ -M <N> ]           Runing memory size as G roughly,
                     default 80% of physical memory
[ -m Hao/InterList/InterSet]    Method for cvtree, defaut: Hao
[ -C ]               Force use the netcdf compress distance matrix
[ -q ]               Run command in queit mode
[ -h ]               Disply this information
```

2.2 Obtain Phylogeny Tree Step by Step

- cv – Generate CV files from input data

```
cv [ -G <dir> ]         input genome file directory
[ -V <dir> ]         output cv directory
[ -i list ]          input species list, defaut: list
[ -f <Fasta> ]       get cv for only one fasta
[ -k '3 4 5 6 7' ]   values of k, defaut: N = 3 4 5 6 7
[ -g faa ]           the type of genome file, defaut: faa
[ -m Hao/InterList/InterSet]    the method for cvtree, defaut: Hao
[ -q ]               Run command in queit mode
[ -h ]               disply this information
```

- dist – Generate distance matrix based on CV files

```
dist
[ -o <dm> ]      Output distance matrix, defaut: <Method><Suffix>
[ -V <cvdir> ]   Super directory of extend cv files
[ -i list ]      Genome list for distance matrix, defaut: list
[ -s <Suffix> ]  Suffix of the cvfile, default: .faa.cv6
[ -r <matrix> ]  Reference distance matrixs, splite with ','
[ -M <N> ]       Runing memory size as G roughly,
                 default 80% of physical memory
[ -m Hao/InterSet/InterList] Method for cvtree, defaut: Hao
[ -C ]           Force use the netcdf compress distance matrix
[ -q ]           Run command in queit mode
[ -h ]           Disply this information
```

- nj – Generate newick tree from distance matrix by using the Neighbor-Joint method

```
nj
[ -d infile ]     Input distance matrix, defaut: dist.matrix
[ -o Tree.nwk ]   Output newick tree, defaut: Tree.nwk
[ -i <list> ]     selection index list of the distance matrix,
                  if no defined, whole distance matrix are used
[ -C ]            Use the netcdf input format, default false
[ -q ]            Run command in queit mode
[ -h ]            Disply this information
```

2.3 Tools

- cvdump – convect the binary cv file to acsii file

```
cvdump  -i <cvfile>        input file name
 [ -g faa|ffn ]      the type of genome file, defaut: faa
 [ -n ]              output the number code, default: the letters
 [ -h ]              disply this information
```

- getdist – get the distance from distance matrix

```
getdist [ -d infile ]       distance matrix files list separate by :, defa
[ -i selfile ]      the select genomes, defaut: none
[ -H ]              whether output html table, default: false
[ -l ]              get the list of species of distance matrix
[ -h ]              disply this information
```

3 The Algorithm

3.1 CVTree scheme

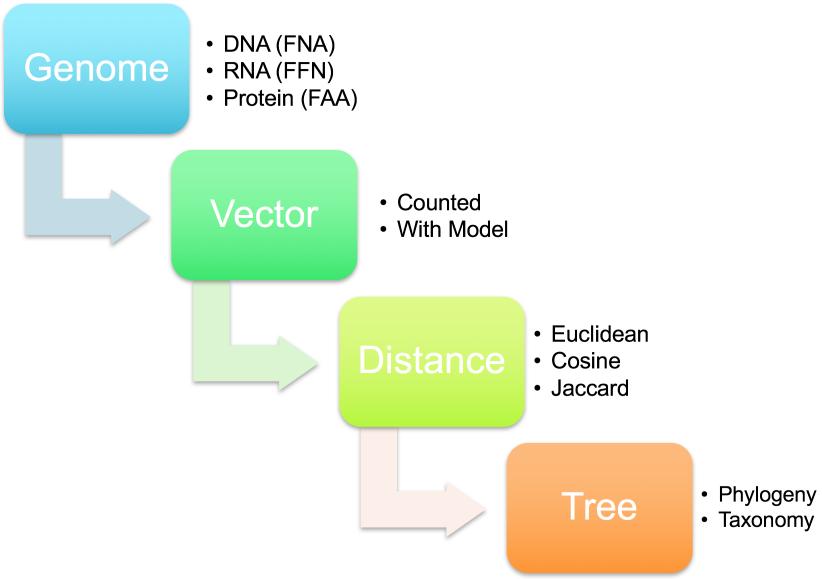

Figure 2: The Scheme of CVTree

3.2 Hao Method: The classical CVTree

The algorithm of CVTree consists of the following steps:

1. Fix a string length K ($K \in [3, 7]$ for Amino Acid sequences and $K \in [3, 16]$ for nucleotide sequences in 32-bit system). Read in the sequence collection of each species separately. Count the number of all K, $K-1$ and $K-2$ tuples for a species. A *raw* Composition Vector (CV) of dimension 4^K or 20^K is formed by putting the counts of K-tuples in lexicographic order.

2. Calculate the subtraction score for the i-th K-tuple:

$$a_i(a_1a_2\cdots a_K) \equiv \frac{f(a_1a_2\cdots a_K) - f^0(a_1a_2\cdots a_K)}{f^0(a_1a_2\cdots a_K)}$$

where $f(a_1a_2\cdots a_K)$ is the frequency of K-tuple, $f^0(a_1a_2\cdots a_K)$ is the frequency predicted from that of $(K-1)$ and $(K-2)$ tuples by using a $(K-2)$-th Markov assumption, [Qi *et al.*, 2004*b*]. All components of

the *raw* CV is replaced by its subtraction score to yield a renormalized CV.

3. Using the renormalized CVs to calculate the pairwise dissimilarity between two species:

$$d(A,B) = (1 - C(C\vec{V}_A, C\vec{V}_B))/2,$$

where

$$C(C\vec{V}_A, C\vec{V}_B) = \frac{\sum_{i=1}^{N} A_i \times B_i}{(\sum_{i=1}^{N} A_i^2 \times \sum_{i=1}^{N} B_i^2)^{\frac{1}{2}}}$$

4. Then obtain the phylogenetic tree (Newick Format) based on this dissimilarity matrix by Neighbor Joint method.

For more detailed description of the algorithm please consult [Qi *et al.*, 2004*b*].

3.3 The update CVTree Algorithm

There are two new algoriths named by InterSet and InterList. The are based on the counted Kstring without the markov model.

3.3.1 The InterSet Method

- From genome to vector:

$$v(a_1a_2\cdots a_k) = \sigma[f(a_1a_2\cdots a_k)]$$

where $\sigma = 1$ when $a_1a_2\cdots a_k$ was found in the genome, or $\sigma = 0$.

- From vector to distance: for the vector i and j,

$$D(\vec{v_i}, \vec{v_j}) = 1 - \sqrt{\frac{\vec{v_i} \cdot \vec{v_j}}{|\vec{v_i}| \cdot |\vec{v_j}}}$$

- Then obtain the phylogenetic tree (Newick Format) based on this dissimilarity matrix by Neighbor Joint method.

3.3.2 The InterList Method

- From genome to vector:

$$v(a_1a_2\cdots a_k) = f(a_1a_2\cdots a_k)$$

where $\sigma = 1$ when $a_1a_2\cdots a_k$ was found in the genome, or $\sigma = 0$.

- From vector to distance: for the vector i and j,

$$D(\vec{v_i}, \vec{v_j}) = 1 - \frac{2 \times \sum \min[v_i(a_1 a_2 \cdots a_k), v_j(a_1 a_2 \cdots a_k)]}{\sum v_i(a_1 a_2 \cdots a_k) + \sum v_j(a_1 a_2 \cdots a_k)}$$

- Then obtain the phylogenetic tree (Newick Format) based on this dissimilarity matrix by Neighbor Joint method.

4 Version History and Contributors

Since 2002 the CVTree software has undergone many revisions. A few major versions were:

1. Web Server CVTree 1.0 was written by Ji Qi, Hong Luo and Bailin Hao
2. Most 1.x versions of Standalone CVTree was written by Lei Gao; Ver. 1.9.6 was written by Ji Qi.
3. Web Server CVTree 2.0 was written by Zhao Xu and Bailin Hao
4. Standalone CVTree 2.4 was written by Zhao Xu
5. Web Server CVTree 3.0 was written by Guanghong Zuo and Bailin Hao
6. Standalone CVTree 3.0 was written by Guanghong Zuo
7. Standalone CVTree 4.0 was written by Guanghong Zuo

5 Citing CVTree in a Publication

Please cite:

1. Ji Qi, Bin Wang, Bailin Hao (2004), Whole proteome prokaryote phylogeny without sequence alignment: a K-string composition approach, *Journal of Molecular Evolution*, **58**: 1 – 11.
2. Guanghong Zuo and Bailin Hao (2015) CVTree3 Web Server for Whole-genome-based and Alignment-free Prokaryotic Phylogeny and Taxonomy. *Genomics Proteomics Bioinformatics,* **13**: 321–331.

References

[Qi *et al.*, 2004*a*] Qi,J., Luo,H. and Hao,B. (2004*a*) Cvtree: a phylogenetic tree reconstruction tool based on whole genomes. *Nucleic acids research,* **32**, W45–7. PMID: 15215347.

[Qi *et al.*, 2004*b*] Qi,J., Wang,B. and Hao,B. (2004*b*) Whole proteome prokaryote phylogeny without sequence alignment: a k-string composition approach. *Journal of Molecular Evolution,* **58**, 1–11.

[Xu and Hao, 2009] Xu,Z. and Hao,B. (2009) Cvtree update: a newly designed phylogenetic study platform using composition vectors and whole genomes. *Nucleic Acids Research,* **37 Web Server Issue**, W174–W178.

[Zuo and Hao, 2015] Zuo,G. and Hao,B. (2015) CVTree3 Web Server for Whole-genome-based and Alignment-free Prokaryotic Phylogeny and Taxonomy. *Genomics Proteomics Bioinformatics,* **13**, 321–331.