# Documentation for VORONOI, version 2.0

Software code by Matthew Stephens[1]
WWW: http://github.com/stephens999/voronoi

Revised by Mary Kuhner[2] January 4, 2021

[1] email: mstephens@uchicago.edu
[2] email: mkkuhner@uw.edu

# Contents

# 1 Introduction

The VORONOI program was written by Matthew Stephens and members of his lab. Mary Kuhner and Jon Yamato in the Wasser Lab have taken over maintenance and expansion of the program, and wrote this documentation. Some of the statements in the documentation are our best deduction from the source code and published paper. If you find any errors please contact us at `mkkuhner@uw.edu`. Locations to obtain the most recent version of the software, and instructions for citing it, are in section 10.

VORONOI is a postprocessor for output from the SCAT2 program (Wasser et al. 2004, see section 10). It requires SCAT2 to have been run in the mode where it assigns the locations of individuals based on their genetic data. VORONOI attempts to refine the estimates by assuming that the location-unknown individuals are likely to be clustered relative to the entire reference set of individuals. It was designed to tighten estimates of the locations of elephant tusks from the same ivory shipment, which may represent a small region within the elephant species range. The algorithm is described in Wasser et al. (2007).

VORONOI is appropriate when there is reason to think that the unknown samples are geographically clustered relative to the reference samples, for example because they were taken in the same shipment, found in the same marketplace, or obtained from the same collecting expedition. It is likely not helpful in other situations.

In our experience, when the majority of individuals in the sample come from a restricted location but there are a few sporadic individuals from elsewhere, VORONOI tends to put the sporadic individuals with the others if there is any support for such a placement at all. The experimental "cutoff" option may possibly help in this situation. Sporadics can be detected by comparing the raw SCAT2 results with the VORONOI results: an individual which is placed in the VORONOI main cluster despite very low SCAT2 support for this placement should be regarded skeptically.

# 2 Compiling VORONOI

VORONOI comes with a rudimentary Makefile. Go to the `src` directory and type "make" to compile it. The executable will be named VORONOI.

As distributed, VORONOI will be compiled in optimized format, which is fastest to run but does not work with a debugger. If you need a debug version, edit the Makefile, commenting out the line that reads:

```
CFLAGS = -O3 $(INCLUDE)
```

and commenting in the line that reads:

```
#CFLAGS = -g $(INCLUDE)
```

Once this is done, type "make clean" followed by "make" to recompile. This will produce a version with debugging information, and will also enable internal correctness checks, which will generally halt the program with a screen error message if they detect a problem. This version is too slow for routine use but can be very helpful for debugging. Please report any bugs you uncover to `mkkuhner@uw.edu`.

To go back to the optimized version, revert to the original Makefile and do "make clean" and "make" again.

For different OS variants you may need to comment in one of the LDFLAGS lines in the Makefile or make further changes. The distributed Makefile is known to work as-is on Ubuntu 18.04 and 20.04.1, and CentOS 7.7.1908.

# 3 Running VORONOI

VORONOI is a command-line program and can be run with the following command:

./VORONOI [options] controlfile PREFIX

The format of `controlfile` is described in the input data section. It is essentially a list of which individuals (by number) are to be analyzed. The `PREFIX` is an arbitrary string (without spaces or hyphens) that will be attached to all of the output files from this run so that they can easily be recognized.

VORONOI also requires appropriately pre-processed SCAT2 output files to be present in the directory where it is invoked, as described below.

# 4 Input data

## 4.1 Pre-processed SCAT2 data

VORONOI currently has a highly specific input format. It requires the output of exactly 9 SCAT2 runs on the same input data set, but with different random numbers. Furthermore, each SCAT2 run must be set to produce eactly 200 samples (of which the first 100 will be discarded as burnin). We recommend setting up 9 directories and running SCAT2 in each.

Once the 9 SCAT2 runs are finished, their output must be consolidated into the directory where VORONOI will be run. SCAT2 writes one location file per individual, named with that individual's ID. To convert to VORONOI input, the 9 groups of SCAT2 results are assigned tags "r" through "z" (lower case). Each individual is given a number; numbers of less than 3 digits are padded on the left with zeros. Thus, the third individual could be coded as 003. (The numbers are arbitrary and need not be consecutive.) Each SCAT2 output file must be copied into the working directory with a new name consisting of that individual's number and the letter for that group: in the case of the third individual, these files would be named 003r, 003s ... 003z. File 003r is the SCAT2 results for the third individual from the first SCAT2 run. (We strongly recommend copying, not renaming, as it reduces the chance you will become confused about which number is which individual.)

### 4.2 Control file

You will also need to prepare a control file which informs VORONOI of the individuals to be analyzed. This is a plain text file containing exactly one line of text. The first number in the file gives the count of individuals to run. After this, on the same line, are the numbers of the individuals to be run (using the numbering scheme used to name the files, except that padding with zeroes is not required). So the following control file:

```
3 1 2 4
```

tells VORONOI to run 3 individuals, numbers 1, 2 and 4. With this input file, VORONOI will read SCAT2 data files 001r...001z, 002r...002z, and 004r...004z. It will ignore all other files that may be present in the directory, allowing you to run different combinations of individuals without having to add or remove files.

## 5 Boundary specification

VORONOI needs to know the boundaries within which individuals could possibly be found. *This boundary specification must match that used in the SCAT2 runs.* As a result, if you have old SCAT2 runs you will not be able to use grid file boundary specification, as this was only added to SCAT2 recently. The boundary can be specified in three ways, with the most useful listed first.

### 5.1 Grid file

Inform the program of the use of a grid file with the -g option:

```
-g gridfilename
```

Internally, VORONOI represents the area under study as a grid with squares 1 degree in latitude and longitude, and aligned on the latitude/longitude lines. The grid file allows the user to specify exactly which grid squares are allowable locations for sampled individuals. A major advantage of the grid file is that it allows the species range to be discontinuous: for example, an organism found on several islands in a chain, but not in the ocean between them.

The grid file should contain the coordinates of each grid square that can contain individuals, one per line, listed as (decimal) latitude and longitude. The latitude and longitude given will be interpreted as the southwest corner of the square, so the northeast corner will have latitude and longitude one greater than given. For example:

```
-11 3
```

specifies that the species can be found in the square whose southwest corner is at 11 S and 3 E, and whose northeast corner is at 10 S and 4 E. It does not matter in what order the squares are listed.

The program will do its inference, internally, on a square grid large enough to hold all of the listed squares plus a minimum 7 degree border around the edges. The border is considered outside the species range and is included to prevent the inference from being perturbed by edge effects.

The code for grid file handling does not handle the poles or the line opposite the Prime Meridian correctly. We recommend modifying the latitudes and longitudes to move them away from this area (don't forget to move them back before interpreting the results). Also, the 1 degree grid "squares" will become far from square in areas close to the poles and this may affect accuracy, although we believe this will be a bigger issue for SCAT2 than VORONOI. All of our practical experience is with African elephants, which occur fairly far from the poles.

## 5.2 Polygon boundary file

This option allows you to specify the boundary of the species range as an arbitrary polygon. The polygon is specified by entering the latitude and longitude of each consecutive vertex into a "boundary" file (see below), and using the `-B` option to tell the software the name of this file. Boundary file use is indicated by the -B option:

```
-Bboundaryfilename
```

Note that (for backwards compatibility) there is no space between the B and the file name. The boundary file should contain one row for each vertex in the polygon, with two numbers on each row (decimal latitude and longitude, with N and E being positive, S and W being negative) separated by a space. The vertices should be entered in order (in either direction around the polygon), with coordinates of the last vertex being an *exact* repeat of the coordinates of the first vertex.

For example, to specify a square region, from 5°$N$ to 3°$S$ and 2°$W$ to 1°$E$ the file could be

```
5 -2
5  1
-3 1
-3 -2
5 -2
```

This code was ported from SCAT2. We prefer use of the grid file as it does not require that the species range be a single contiguous polygon. While the boundary file approach may appear capable of more precision, VORONOI operates on a grid of one degree squares internally and cannot make use of the additional precision.

The routine that tests whether a point is inside or outside the specified polygon is based on the 2-D algorithm described by Dan Sunday (`http://www.softsurfer.com`). It will not work for regions that span 180°$E$ (the line opposite the prime meridian) unless you enter the Longitudes as having the same sign, eg by using 190 and 170 instead of $-10$ and 170. It also will not work for regions including the North or South Pole. We recommend recoding such data (and reversing the recoding before interpreting the results). It is possible that no usable recoding exists for a circumpolar or very widespread species.

### 5.3 Inbuilt boundaries

VORONOI was written to analyze African elephant data, and contains hard-coded boundaries for savannah and forest African elephants. These boundaries are imperfect–the savannah boundary contains large areas of ocean–and we do not recommend their use for anything other than deliberate replication of previous results.

The savannah hard-coded boundaries can be accessed with `-d`, and the forest boundaries with `-D`. No auxillary input file is needed. Consult the output *PREFIX_mapinfo* file to see details of the grid used.

## 6 Other useful options

The following options are specified on the VORONOI command line.

### 6.1 Setting the random number seed: `-S`

The `-S` option allows you to specify the random number seed, which should be a positive integer. If no seed is set, one will be derived from the system clock. Setting the seed is useful to allow you to replicate a run: for example, debugging is much easier if you are running the same execution path each time. It is also useful when multiple copies of the program are being launched by a script, as if they are launched too quickly they may get the same clock time and thus the same seed.

### 6.2 Alternative output format: `-v`

The verbose option `-v` will cause VORONOI to write an additional output file presenting the same results as the indprobs file, but differently formatted (longer but easier to parse). For details see the output section.

## 7 Experimental options

### 7.1 Likelihood cutoff

VORONOI normally considers a clustering solution valid if it explains at least one SCAT2 observation for each individual. This can lead to over-clustering: if there is a tight cluster of many individuals, and the remaining individuals have very weak but non-zero support for falling in the cluster, VORONOI is liable to put all of the individuals there.

This behavior can be changed with the cutoff `-C` option. Follow the option with a decimal number between zero and one inclusive. Only solutions which explain more than this proportion of results for each individual will be considered valid. A cutoff of zero emulates the previous behavior of VORONOI and is equivalent to having no cutoff at all. A cutoff of one means that all SCAT2 results for every individual must be included in the VORONOI solution, which in practice means that VORONOI will nearly replicate the SCAT solutions. Very limited work has been done to find whether setting a non-zero cutoff improves VORONOI results, but it is clear that appropriate cutoffs are likely to be quite small: in our test data, cutoffs greater than 0.2 produce results nearly indistinguishable from the underlying SCAT2 results, and the best results seen are for much lower but non-zero cutoffs such as 0.02.

# 8 Outputs

VORONOI writes several files. Each filename will begin with the PREFIX specified in the run command, so that results from the same run can be easily recognized.

## 8.1 Map info file

This file is named *PREFIX_mapinfo*, and contains information for interpreting the VORONOI map grid: this is essential for correct interpretation of the results. VORONOI works in 1 degree latitude and longitude squares, and refers to each square by the decimal latitude and longitude of its south-western corner. Thus, the grid square at -14, 21 (14 degrees South, 21 degrees East) extends from -14,21 to -13,22. It also gives the number of squares in the grid, which can be helpful in post-processing the results. Currently the grid must be square, though we plan to remove this restriction in a later release.

## 8.2 Indprobs file

This file is named `PREFIX_indprobs`, and contains the posterior probability for each grid location, for each sample in turn. If the grid is, for example, 67 x 67, the first 67 lines will correspond to the first sample listed in the input file, the second 67 lines will correspond to the second sample, and so forth. Each entry gives the posterior probability that the given individual is found in the given grid square.

Note that the order of entries is the order in the input file; if the input file is out of numerical order, the output will be too (this is not recommended!) You will want to keep very careful track of which individual corresponds to each entry. A future project is having VORONOI read SCAT2 input directly and retain the sample IDs, to avoid problems of this kind.

Note that the first grid square written out is the LOWER LEFT grid square, even though it will fall in the upper left part of the printout. If your results seem upside down, it's likely because you are interepreting the results as if 0,0 was the square in the UPPER LEFT, which is visually suggested by the way the output is printed. If you find this confusing (we do) we recommend turning on the `-v` option, which writes the output in a different format.

The maximum entry for a sample can be taken as a point estimate of its location, but we recommend looking at the whole grid. A useful way to do so is to plot it as a heatmap onto a world map of the area in question. VORONOI placements can consist of several well separated plausible regions, and a single point estimate does not represent such data well.

## 8.3 Printprobs file

This file is only written if the verbose `-v` option is given, and is named `PREFIX_printprobs`. It contains exactly the same information as `PREFIX_indprobs`, but presented in a different way. Roughly, indprobs is good for spreadsheet users and R programmers, printprobs is good for other users.

In the printprobs file each individual in turn is represented by a number of lines equalling the number of squares in the grid. Each line gives the decimal latitude and longitude of a grid square and the posterior probability for that individual to be in that grid square. Thus, it is not necessary to convert between grid square index and latitude/longitude coordinates in order to interpret this file.

## 8.4 Regions file

This file is named `PREFIX_regions`. It contains one grid per non-burnin iteration of VORONOI. These grids are dimensioned in the same way as the ones in `PREFIX_indprobs` but contain only 1's and 0's. A square is 1 if it is both within the species range and within VORONOI's current estimate of the region containing all the samples (called "R" in the published paper). Thus, one could plot these to see how R evolved over the course of the run.

## 8.5 Overall probabilities file

This file is simply named *PREFIX*. It contains one grid, dimensioned the same way as the ones in `PREFIX_indprobs`, but containing the posterior probability, for each grid square, that it was live (within both the region R in which all samples were inferred to lie, and the species range), taken as an average over all iterations. Surprisingly, this includes burn-in iterations. The contents of this file can be taken as an estimate of the region R.

## 8.6 Trace file

This file is named `PREFIX_trace`. It contains two columns: the first is iteration number, and the second is the value, at that iteration, of the metaparameter `p`. This parameter estimates what proportion of Voronoi polygons will fall within the species range. The file can be examined to see if the search over this parameter is mixing well, and what the range of inferred values of `p` is. A useful tool for this purpose is the TRACER program (see 10). All iterations are listed, including burnin. If an estimate of `p` is desired, the mean or median of the non-burnin iterations are reasonable estimates.

If, for a given iteration, the posterior probability for that iteration is calculated as less than or equal to zero, or greater than or equal to one, no entry is made in the trace file for that iteration. On our data this is less than 1% of iterations. Such iterations are always rejected, so if a full trace is desired, missing entries can be replaced by the entry immediately before them. This is likely to be a bug but seems fairly harmless.

# 9 Changes between 1.0 and 2.0

The original VORONOI code did not have a version number but will be called version 1.0.

## 9.1 Bugs

*Coherency bugs.* Two serious bugs present in 1.0 were corrected in 2.0. Both caused incoherency of the program state on a particular accepted or rejected MCMC step. Specifically, the program maintained both a list of which polygons were active, and a count of how many grid squares were in active polygons. On some MCMC steps the count of grid squares was not correctly updated and would disagree with the information in the list of polygons. We believe that these bugs caused a bias towards tighter clustering (fewer active polygons) but their main effect was to add noise to the analysis. We recommend re-running any results obtained with 1.0.

*Inference of $p$.* The mathematical model was changed between 1.0 and 2.0. The 1.0 behavior may have been intended but we believe it to be incorrect. Specifically, the metaparameter $p$ gives the probability that a particular Voronoi polygon will be included within the putative region R from which the samples were drawn. In version 1.0, polygons which fell totally outside the species range, and could thus never be included in R, were scored when proposing changes to $p$. This caused a downwards bias in $p$, especially if the species range was much smaller than the VORONOI grid, and this bias led to over-clustering (a too-small R). In 2.0, polygons outside the species range are not considered in changing $p$. In elephant data we have found this to improve accuracy, evaluated by estimating the location of individuals for whom the sampling location is known.

*Disagreement between SCAT2 and VORONOI.* When SCAT2 and VORONOI are run with either the hardcoded range boundaries, or range boundaries set as a polygon bounding box, they can disagree over the legality of an individual's placement: SCAT2 asks whether the individual's exact placement is in-bounds, whereas VORONOI asks whether the center of the 1 degree grid square in which that placement falls is in-bounds. Version 1.0 ignored this issue: we are not sure what the effect of these out-of-bounds results was on the resulting estimates. In version 2.0, SCAT2 results which fall slightly outside the VORONOI boundary are moved into an adjacent in-boundary square; if no such square is available the program will terminate. (In our experience, if the program terminates for this reason you have probably used different boundaries in SCAT2 and VORONOI and need to correct that.) This issue does not arise with use of grid square range boundaries as the identical rule for legality is then used in both programs.

It should also be noted that since the species boundary must match in SCAT2 and VORONOI, and VORONOI 1.0 did not accomodate boundary files, all runs made using a boundary file in SCAT2 and then analyzing the results with VORONOI 1.0 were incorrect. The likely behavior is that SCAT2 results falling outside VORONOI's boundaries were silently discarded, reducing statistical power. (The program would crash if all of the SCAT2 results for a given individual were outside the VORONOI boundaries, but even one in-range SCAT2 result would let the run proceed.)

## 9.2 Additions

This manual was added to supplement the original "readme" file.

Version 1.0 could not analyze more than 999 samples due to requiring a three-digit ID code. Version 2.0 can analyze an arbitrary number of samples. Zero-padding to exactly three digits is still required for 1 or 2 digit ID codes.

Version 1.0 was specialized to work on savannah or forest elephants and had hard-coded species boundaries appropriate to these taxa. Version 2.0 can accept boundary information (as either a set of polygon vertices or a set of grid square coordinates) for any desired species range as long as it does not include either pole or cross the line opposite the Prime Meridian. **It is essential to use the same species boundary for the SCAT2 runs and the VORONOI run.** Use the `-b` option for a set of polygon vertices or the `-g` option for a set of grid square coordinates.

Version 1.0 always used a 67 x 67 grid of one-degree squares. Version 2.0 uses this grid if the built-in defaults or a polygon boundary file are used, but if a grid file is used, VORONOI will instead determine the size of its grid by taking the smallest square region that contains all of the specified grid squares and then adding a 7 square boundary on all sides.

Limited error checking of input files and data has been added.

Two new command line options were added: verbose output (`-v`) and the experimental cutoff option (`-C`).

Two new output files were added: the `PREFIX_mapinfo` file gives information on the size and geographic location of the VORONOI grid, and the optional `PREFIX_printprobs` file repeats the information from `PREFIX_indprobs` in a more verbose format.

## 10 Software and publications

When using this program, *please specify the version of the software you used,* and cite the following publications:

VORONOI

Source code: `http://github.com/stephens999/voronoi`

Publication:

Wasser SK, Mailand C, Booth R, Mutayoba B, Kisamo E, Clark B, Stephens M (2007) Using DNA to track the origin of the largest ivory seizure since the 1989 trade ban. PNAS 104: 4228-4233.

SCAT2

Source code: `http://github.com/stephens999/scat`

Publication:

Wasser SK, Shedlock AM, Comstock K, Ostrander EA, Mutayoba B, Stephens M (2004) Assigning African elephant DNA to geographic region of origin: applications to the ivory trade. PNAS 101: 14847-14852.

A third-party tool that can be useful with SCAT2 and VORONOI is the TRACER utility:

TRACER

Source code: `https://beast.community/tracer`

Publication:

Rambaut A, Drummond AJ, Xie D, Baele G, Suchard MA (2018) Posterior summarization in Bayesian phylogenetics using Tracer 1.7. Syst Biol. syy032. doi:10.1093/sysbio/syy032.