APL Pipeline

These scripts are written in APL for the APL+Win Version 2.0.00 software. Because APL commands can be executed individually, as well as by script, some of the steps in our analyses were done with individually executed steps rather than with scripts. When this was done, it is explained below. Scripts are listed in APLPLUS font, whereas descriptive material is in Times New Roman (this font).

I. Reading SNPs from file JG1 (in script READCALLS) and SNP indexing

A. The script READCALLS produces 3 variables:

1. ΔSCORES, an N X 62 X 2 variable. Each 62 X 2 submatrix holds the SNP alleles for 62 individuals.
2. ΔTIGS, a vector with contig number corresponding to each SNP
3. ΔPOS, a vector with contig position corresponding to each SNP.

It reads calls from a Table file that was produced using the GATK VariantsToTable tool. In the script, this file is referred to as 'JG1'.

```
**********
READCALLS

⍝ THIS PROGRAM READS IN DATASET JG1 AND CALCULATES ALLELE COUNTS SEP
ARATELY FOR
⍝    THE DIFFERENT POPULATIONS
⍝ ALSO, MAKES THREE VARIABLES:
ΔSCORES←0 62 2⍴' '  ⍝ VARIABLE TO HOLD GENOTYPE CALLS
ΔTIGS←0⍴0           ⍝ VARIABLE TO HOLD CORRESPONDING CONTIG NUMBERS
ΔPOS←0⍴0            ⍝ VARIABLE TO HOLD CORRESPONDING GENOME POSITION

'C:\JG1' ⎕NTIE ¯1
SIZE←⎕NSIZE ¯1
FILE←⎕NREAD (¯1,82,(SIZE+1000),0)   ⍝  READ IN FILE
⎕NUNTIE ¯1
HEADER←843↑FILE     ⍝   SEPARATE HEADER
DATA←843↓FILE       ⍝   REMOVE HEADER FROM DATA
SAMPLES←79↓HEADER
DATA2←DATA,'tig'    ⍝   APPEND 'TIG' TO END OF DATA
CHET←LHET←AUSTHET←WHET←0⍴0    ⍝ INITIALIZE VARIABLES FOR INDIVIDUAL
SAMPLE HETEROZYGOSITIES
TOTC←TOTL←0 5⍴0               ⍝ SET UP VARIABLES FOR TOTAL COUNTS FO
R C AND L FOR 5 NUCLEOTIDES (INCLUDING ⋆)

CINDEX←(1+⍳31),62    ⍝  SET UP INDICIES FOR ROWS OF DATA2 TO DISTING
UISH SAMPLES FROM DIFFERENT POPULATIONS
AUSTINDEX←1
LINDEX←32+⍳29


I←0
RETI:I←I+1
```

```
TEMP6←((I÷1000)-(⌈(I÷1000)))    ⍝ REPORT SNP NUMBER EVERY 100 SNPS
⍎(TEMP6=0)/'⎕TCLF ⋄ I'

⍝  PICK DATA FOR NEXT SNP--STRIP OFF LEADING INFO AND LEAVE JUST NUC
LEOTIDES FOR DIFFERENT
⍝       SAMPLES.  TEMP5 HAS FORMAT G/G G/G G/C . . .
TEMP←5000↑ DATA2
TEMP2←TEMP ⎕SS 'tig'
TEMP3←TEMP2/⍳⍴TEMP2
TEMP4←(TEMP3[2]-1)↑TEMP
TEMP5←(TEMP3[1]-1)↓TEMP4

⍝  IDENTIFY NUCLEOTIDE IN REFERENCE GENOME (NOT CURRENTLY USED^
IND0←(TEMP5=⎕AV[10])/⍳⍴TEMP5
TIG←IND0[1]↑TEMP5
TIG2←(TIG∈'0123456789')/TIG
TIG3←⎕FI TIG2
ΔTIGS←ΔTIGS,TIG3

POS←¯1↓(IND0[1]↓IND0[2]↑TEMP5)
POS2←⎕FI POS
ΔPOS←ΔPOS,POS2

⍝  REFORMAT TEMP5 AS 62 X 3 MATRIX OF NUCLEOTIDES IN FORMAT G/G
IND1←TEMP5='/'
IND2←IND1/⍳⍴IND1
SCORES←(IND2[1]-2)↓TEMP5
SCORES2←62 4⍴SCORES
SCORES3←SCORES2[;⍳3]
CHECK←(+/SCORES3[;2]='/')=62     ⍝ CHECK TO ENSURE PROGRAM IS NOT OUT
 OF ALIGNMENT
⍎(CHECK=0)/''''BAD DATA FOR SNP '',I ⋄ ⎕TCLF ⋄ SCORES3 ⋄→0'
ΔSCORES←ΔSCORES,[1]SCORES3[;1 3]

DATA2←(TEMP3[2]-1)↓DATA2        ⍝  DROP CURRENT SNP FROM DATA2

TEST←+/(DATA2 ⎕SS 'tig')

→(TEST>1)/RETI
⎕TCLF
'PROGRAM COMPLETE'
'     HETEROZYGOSITIES IN VARIABLES CHET, LHET, AUSTHET, AND WHET'
'     NUCLEOTIDE COUNTS IN VARIABLES TOTC AND TOTL'
'     CALLS IN VARIABLE ''ΔSCORES'''
'     CORRESPONDING CONTIGS IN VARIABLE ''ΔTIGS'''
'     CORRESPONDING CONTIG POSITION IN VARIABLE ''ΔPOS'''
'NOTE: THESE VARIABLES SHOULD BE SAVED TO AN APL COMPONENT FILE'
**********************
```

A series of index variables are created manually. These are vectors corresponding to rows in the 62 X 2 submatrices of ΔSCORES that correspond to particular subsets of samples. For example, ΔCINDEX and ΔLINDEX correspond to *I. cordatotriloba* and *I. lacunosa* samples. ΔCALLO3 and ΔLALLO3 correspond to known allopatric samples for the two species. \

B. A series of scripts is run to identify codon and codon position of SNPs using the LAC genome and produce a number of indexing variables. The following is an overview of these scripts and how they are used:

1. PROGRAM 'READLACGFF' READS IN THE I. LACUNOSA GFF3 FILE AND CONVERTS IT TO A GFF3-LIKE MATRIS ('ΔLACCDS') THAT KEEPS ONLY LINES WITH COLUMN 3 = 'CDS'. 'ΔLACCDS IS SAVED IN THE COMPONENT FILE 'C:\LACCDS'

2. 'ΔLACCDS' IS REORDERED IN ORDER OF SCAFFOLD NUMBER AND PUT IN 'ΔLACCDS2'. WITHIN A GIVEN SCAFFOLD, THE START AND END POSITIONS OF THE CDS ELEMENT ARE NOT ORDERED

3. 'CONVLAC' IS USED TO REORDER BEGINNING POSITIONS WITHIN SCAFFOLDS AND ELIMINATE DUPLICATE ENTRIES. IT PRODUCES 'ΔLACCDS4'

THE COLUMNS OF 'ΔLACCDS4' ARE:

COLUMN1	CONTIG NO.
COLUMN2	STARTPOS IN CONTIG
COLUMN3	ENDPOS IN CONTIG
COLUMN4	STRAND
COLUMN5	PHASE

4. 'FIXLACCDS4' IS USED TO ELIMINATE ADDITIONAL DUPLICATES FROM 'ΔLACCDS4' AND PRODUCE 'ΔLACCDS5'

THE COLUMNS OF 'LACCDS5' ARE THE SAME AS FOR 'ΔLACCDS4'

5. RUN 'INDEXLAC' TO MAKE INDEX OF SCAFFOLDS THIS PROGRAM INDEXES THE SCAFFOLDS OF THE I. LACUNOSA GENOME. IT CREATES THE VARIABLE 'ΔLACINDEX', WHICH HAS THE FOLLOWING COLUMNS:

COLUMN1	SCAFFOLD NUMBER
COLUMN2	START POSITION OF SCAFFOLD IN GENOME FASTA FILE
COLUMN3	END POSITION OF SCAFFOLD IN GENOME FASTA FILE

6. 'GETLACSNPS2' READS IN SNPS FROM A TABLES FILE AND MERGES THE INFORMATION WITH 'Δ LACCDS5' TO PRODUCE THE VARIABLE 'ΔLACSNPS', WHICH HAS THE FOLLOWING INFORMATION:

COLUMN1	CONTIG NO.
COLUMN2	START POSITION OF CDS FEATURE
COLUMN3	POSITION OF SNP
COLUMN4	END POSITION OF CDS FEATURE
COLUMN5	STRAND (0 = -, 1 = +)
COLUMN6	PHASE (CODON POSITION (0,1,2) OF START POSITION OF CDS FEATURE
COLUMN7	ALLELE 1
COLUMN8	ALLELE 2

'GETLACSNPS2' ALSO MAKES 'ΔLACCODONS', WHICH HAS THE FOLLOWING COLUMNS:

COLUMN1	ALTERNATIVE CODON 1
COLUMN2	ALTERNATIVE CODON 2
COLUMN3	WHETHER DIFFERENCE BETWEEN CODONS IS SYNONYMOUS (S) OR NON-

SYNONYMOUS (N)
COLUMN4 SNP NUMBER

'ΔLACSNPS' AND 'ΔLACCODONS' ARE STORED IN COMPONENT FILE 'LACNPS'

7. AFTER RUNNING GETLACSNPS2, RUN 'SEPSNPS' TO MAKE FOLLOWING VARIABLES:

'ΔSYNSCORES'	(FROM 'ΔSCORES')	SNP DATA ON JUST SYNONYMOUS SNPS
'ΔNONSCORES'	(FROM 'ΔSCORES')	SNP DATA ON JUST NON-SYNONYMOUS SNPS
'ΔOSCORES'	(FROM 'ΔSCORES')	DATA ON JUST 'OTHER' SNPS

8. RUN 'SPLICE4' TO CALCULATE NUMBER OF SYN AND NON-SYN SITES. PRODUCES TWO VALUES OF COUNTS:

'ΔTOTSYN' AND 'ΔTOTNONSYN'.

SCRIPTS:

```
**************************
READLACGFF
⍝ THIS PROGRAM READS IN PARTS OF THE I. LAC GFF3 FILE IN 10000000 BY
TE CHUNKS
⍝ FOR EACH CHUNK, IT ASCERTAINS WHERE THE CDS FEATURES AND SAVES THE
 LINE GIVING
⍝  INFO OF THAT FEATURE IN VARIABLE NEWGTF2.
⍝  PROCESSING IS AS FOLLOWS:
⍝     1.  FIRST CHUNK IS READ IN.
⍝     2.  ALL LINES WITH FEATURE = 'CDS' ARE KEPT AND APPENDED TO 'N
EWGTF2'
⍝     3.  'NEWGTF2' IS SAVED TO A COMPONENT OF FILE 'GFF3.SF'
⍝     4.  CHANGE J←0 TO J←1 IN PROGRAM AND COMMENT OUT 'GTF←0⍴0' AND
 RESTART
⍝     5.  NEXT CHUNK IS READ IN AND STEPS 2 AND 3 ARE REPEATED,
⍝     6.  STEPS 4 AND 5 REPEATED FOR ALL CHUNKS
⍝     7.  THE DIFFERENT MATRICES (DIFFERENT NEWGTF2) IN 'GFF3.SF' AR
E MANUALLY CATENATED TO
⍝            FORM MATRIX '⍙LACCDS' AND SAVED TO 'GFF3.SF' AS ANOTHER
 COMPONENT
⍝     8.  '⍙LACCDS' IS MANUALLY SORTED BY SCAFFOLD NUMBER TO FORM '⍙
LACCDS2', WHICH IS
⍝            SAVED TO ANOTHER COMPONENT OF 'GFF3.SF'
FLAG←0
J←4
RETJ:J←J+1
DIM←⍴GTF
'C:\LACGFF1' ⎕NTIE ¯1
GTF←GTF,⎕NREAD ¯1 82 10000000 ((J-1)×10000000)
⎕NUNTIE ¯1
⎕TCLF
'SEGMENT ',J,' READ'
```

```
⍎((10000000+DIM)>⍴GTF)/'FLAG←1 ⋄ ⎕TCLF ⋄ ''FLAG SET TO 1'''

 ⍝ NEWGTF2←NEWGTF←0 82⍴' '
 IND←(GTF=⎕TCLF)/⍳⍴GTF
 GTF←IND[1]↓GTF

 I←0
 RETI:I←I+1

 TEST←(I÷100)=(⌈(I÷100))
 ⍎(TEST=1)/'''J = '',J,''  I= '',I,'' MBYTES LEFT: '',(10 5⍕(⍴GTF)÷1
000000)'
 IND←(GTF=⎕TCLF)/⍳⍴GTF
 →(0=⍴IND)/0
 LINE←IND[1]↑GTF
 LINE←¯1↓LINE
 IND2←(LINE=⎕AV[10])/⍳⍴LINE
COL3←¯1↓(IND2[2]↓(IND2[3]↑LINE))
 TEST←∧/COL3[⍳3]='CDS'
 →(TEST=0)/DOWN

 COL1←¯1↓(IND2[1]↑LINE)
 COL2←¯1↓(IND2[1]↓(IND2[2]↑LINE))

COL4←¯1↓(IND2[3]↓(IND2[4]↑LINE))
COL5←¯1↓(IND2[4]↓(IND2[5]↑LINE))
COL6←¯1↓(IND2[5]↓(IND2[6]↑LINE))
COL7←¯1↓(IND2[6]↓(IND2[7]↑LINE))
COL8←¯1↓(IND2[7]↓(IND2[8]↑LINE))
COL9←IND2[8]↓LINE
COL3A←15↑(COL3,15⍴' ')
COL9A←40↑(COL9,40⍴' ')
COL4A←10↑(COL4,10⍴' ')
COL5A←10↑(COL5,10⍴' ')
NEWLINE←COL1,'  ',COL2,'  ',COL3A,'  ',COL4A,'  ',COL5A,'  ',COL6,'
 ', COL7,'  ',COL8,'  '
NEWGTF2←NEWGTF2,[1]NEWLINE

DOWN:IND←(GTF=⎕TCLF)/⍳⍴GTF
GTF←IND[1]↓GTF
→((FLAG=0)∧(0=⍴IND))/RETJ
→RETI

⍝ ⍙LACCDS←NEWGTF2

⎕TCLF
'PROGRAM READLACGFF COMPLETE.  CDS DATA IN VARIABLE ⍙LACCDS'
'THIS VARIABLE NEEDS TO BE MANUALLY REORDERED


*****************************
```

```
CONVLAC
⍝ THIS PROGRAM CONVERTS THE CHARACTER VARIABLE ΔLACCDS2 TO THE NUMER
IC VARIABLE ΔLACCDS3
⍝  SORTS IT AND ELIMINATES DUPLICATE ENTRIES, PRODUCING THE VARIABLE
 ΔLACCDS4
ΔLACCDS3←0 5⍴0
DIM←1↑⍴ΔLACCDS2
⎕TCLF
DIM
START←0

I←0
RETI:I←I+1
TEST←(I÷1000)=(⌈(I÷1000))
⍎(TEST=1)/'I'
LINE←ΔLACCDS2[I;]
CONTIG←⎕FI LINE[3+⍳8]
STARTPOS←⎕FI LINE[49+⍳11 ]
ENDPOS←⎕FI LINE[61+⍳12]

STRAND←0
⍎(LINE[77]='+')/'STRAND←1'
PHASE←⎕FI LINE[80]
NEWLINE←CONTIG,STARTPOS,ENDPOS,STRAND,PHASE
ΔLACCDS3←ΔLACCDS3,[1]NEWLINE
→(I<DIM)/RETI

⎕TCLF
'STARTING CONVLAC2'
CONVLAC2
⎕TCLF
'STARTING CONVLAC3'
CONVTRIF3
ΔLACCDS4←ΔLACCDS3[INDEX9;]
⎕TCLF
'PROGRAM FINISHED.  DATA ON I. LACUNOSA CODING SEQUENCES IN VARIABLE
 ''ΔLACCDS4''.'
*****************************
FIXLACCDS4
⍝  THIS PROGRAM ELIMINATES FURTHER DUPLICATES IN ΔLACCDS4 AND OVERLA
PPING CDS FEATURES

ΔLACCDS5←0 5⍴0

DIM←⍴ΔUNIQSCAFFS
DIM1←1↑⍴ΔLACCDS4

⍝ I LOOP LOOPS THROUGH SCAFFOLDS
I←0
RETI:I←I+1
TEST←(I÷10)=(⌈(I÷10))
⍎(TEST)/'I'
SCAFF←ΔUNIQSCAFFS[I]
IND←(ΔLACCDS4[;1]=SCAFF)/⍳DIM1
PART←ΔLACCDS4[IND;]
```

```
PART←PART[⍋PART[;2];]
DIMPART←1↑⍴PART
TEST←1=DIMPART
⍎(TEST)/'⍙LACCDS5←⍙LACCDS5,[1]PART ⋄ →DOWN2'

⍝ J LOOP REMOVES CDS FEATURES WITH DUPLICATE BEGINNING POSITIONS
DUPS←0⍴0
DIM2←1↑⍴PART
KEEP←0 5⍴0
J←0
RETJ:J←J+1

LINE1←,PART[J;]
LINE2←,PART[J+1;]

⍎(LINE1[2]≠LINE2[2])/'KEEP←KEEP,[1]LINE1 ⋄ →DOWN1'
DUPS←DUPS,J
IND←(PART[;2]=LINE1[2])/⍳DIM2

NEQUAL←⍴IND
LINES←PART[IND;]

IND2←1↑((⌈/LINES[;3])=LINES[;3])/⍳⍴IND
ADDLINE←LINES[IND2;]
KEEP←KEEP,[1]ADDLINE
J←J+(NEQUAL-1)

DOWN1: →(J<(DIM2-1))/RETJ

⍝ K LOOP REMOVES DUPLICATES WITH SAME END POSITION

KEEP←KEEP[⍋KEEP[;3];]  ⍝  REORDER KEEP IN ASCENDING ORDER OF END POS
ITIONS

DUPS2←0⍴0
DIM3←1↑⍴KEEP

⍎(DIM3=1)/'⍙LACCDS5←⍙LACCDS5,[1]KEEP ⋄ →DOWN2'
KEEP2←0 5⍴0
K←0
RETK:K←K+1

LINE1←,KEEP[K;]
LINE2←,KEEP[K+1;]

⍎(LINE1[3]≠LINE2[3])/'KEEP2←KEEP2,[1]LINE1 ⋄ →DOWN3'
DUPS2←DUPS,K
IND←(KEEP[;3]=LINE1[3])/⍳DIM3

NEQUAL←⍴IND
LINES←KEEP[IND;]

IND2←1↑((⌈/LINES[;3])=LINES[;3])/⍳⍴IND
ADDLINE←LINES[IND2;]
KEEP2←KEEP2,[1]ADDLINE
```

```
K←K+(NEQUAL-1)
DOWN3:→(K<DIM3-1)/RETK

ΔLACCDS5←ΔLACCDS5,[1]KEEP2

DOWN2:→(I<DIM)/RETI
**************************
INDEXLAC ;STARTBYTE
⍝ THIS PROGRAM INDEXES THE SCAFFOLDS OF THE I. LACUNOSA GENOME.  IT
CREATES THE VARIABLE
⍝    ΔLACINDEX, WHICH HAS THE FOLLOWING COLUMNS:
⍝        COLUMN1      SCAFFOLD NUMBER
⍝        COLUMN2      START POSITION OF SCAFFOLD IN GENOME FASTA FILE
⍝        COLUMN3      END POSITION OF SCAFFOLD IN GENOME FASTA FILE

STARTBYTE←0
I←0
RETI:I←I+1

TEST←(I÷100)=(⌈(I÷100))
⍎(TEST=1)/'I'

'C:\LACGENOME' ⎕NTIE ¯1
PARTSCAFF←⎕NREAD ¯1 82 2000000 STARTBYTE
⎕NUNTIE ¯1

DIM←⍴PARTSCAFF
→(DIM=0)/0

IND←(PARTSCAFF ='>')/⍳⍴PARTSCAFF
ENDSCAFF←IND[2]-1

TEMPSCAFF←ENDSCAFF↑PARTSCAFF
IND←(TEMPSCAFF=⎕TCLF)/⍳⍴TEMPSCAFF
HEAD←IND[1]↑TEMPSCAFF
SEQ←IND[1]↓TEMPSCAFF
SEQ←(SEQ≠⎕TCLF)/SEQ
ΔSCAFF←⎕FI HEAD[4+⍳8]

SEQ[⍳10]
READLACSCAFF ΔSCAFF
ΔSEQUENCE[⍳10]
TRSH←⎕

⍝  ΔLACINDEX←ΔLACINDEX,[1](ΔSCAFF,(STARTBYTE),(STARTBYTE+ENDSCAFF))

STARTBYTE←STARTBYTE+ENDSCAFF

→RETI
**************************
```

```
GETLACSNPS2

⍝ THIS PROGRAM READS IN SNPS FROM DATASET OF SNP CALLS (I. LAC REFER
ENCE; E.G. JG1-LIKE) AND
⍝    COMBINES WITH DATA FROM ΔLACCDS5 TO FORM ΔLACSNPS, WHICH HAS FO
LLOWING COLUMNS:
⍝
⍝           COLUMN1  CONTIG
⍝           COLUMN2  START POSITION OF CDS FEATURE
⍝           COLUMN3  POSITION OF SNP
⍝           COLUMN4  END POSITION OF CDS FEATURE
⍝           COLUMN5  STRAND  (0 = -, 1 = +)
⍝           COLUMN6  PHASE  (CODON POSITION (0,1,2) OF START POSITION
 OF CDS FEATURE
⍝           COLUMNS7-9  CODONINDEX (POSITIONS OF CODON CONTAINING SNP
; ON + STRAND)
⍝           COLUMN10 ΔSNPNUM (SNP NUMBER, AS COUNTED BY THIS PROGRAM)

ΔLACSNPS←0 10⍴0
ΔLACCODONS←0 19⍴' '    ⍝ THIS VARIABLE CONTAINS CODON INFORMATION FOR
 THE CORRESPONDING SHP
                       ⍝    COMUMN1  ALTERNATIVE ALLELES
                       ⍝    C0LUMN2: ALTERNATIVE CODON 1
                       ⍝    COLUMN3: ALTERNATIVE CODON 2
                       ⍝    COLUMN4: SYNONYMOUS(S) OR NON-SYNONYMOUS
(N) DIFFERENCE
ΔTSCORES←0 62 2⍴' '    ⍝  THIS VARIABLE CONTAINS ALL OF THE SNP CALLS
 IN SAME ORDER AS ΔLACSNPS AND ΔLACCODONS
PROBLEMS←0 3⍴0
NOTFOUND←0⍴0
ΔSNPINFO←0 3⍴0
ΔALTALLELES←0 2⍴' '
ΔUNEQUAL2←0

'C:\JG1' ⎕NTIE ¯1
SIZE←⎕NSIZE ¯1
FILE←⎕NREAD (¯1,82,(SIZE+1000),0)    ⍝  READ IN FILE
⎕NUNTIE ¯1
HEADER←846↑FILE      ⍝   SEPARATE HEADER
DATA←846↓FILE        ⍝   REMOVE HEADER FROM DATA
⍝SAMPLES←79↓HEADER
DATA2←DATA,'tig'     ⍝   APPEND 'TIG' TO END OF DATA

⍝ THE FOLLOWING STATEMENT IS CURRENTLY ACTIVE IN 'ANALYSNPS'
⍝    WHEN THIS PROGRAM IS SET UP TO CALL 'ANALYSNPS', THE STATEMENT
SHOULD BE
⍝    DE-ACTIVATED IN 'ANALYSNPS' AND ACTIVATED HERE

⍝ ΔSNPS←0 4⍴0     ⍝ THIS VARIABLE WILL CONTAIN INFORMATION ON SNP COD
ON POSITION
              ⍝  COLUMNS ARE: (1) SCAFFOLD NUMBER (2) POSITION ON SC
AFFOLD  (3) CODON POSITION (1,2,OR 3)  (4) SYNONYMOUS (0) OR NON-SYN
ONYMOUS (1)
```

```
I←0
RETI:I←I+1

TEST←(I÷100)=(⌈(I÷100))     ⍝ REPORT SNP NUMBER EVERY 100 SNPS
⍞(TEST=1)/'I'

ΔSNPNUM←I      ⍝ SNIP NUMBER

→(ΔSNPNUM>MAXSNP)/0

⍝  PICK DATA FOR NEXT SNP--STRIP OFF LEADING INFO AND LEAVE JUST NUC
LEOTIDES FOR DIFFERENT
⍝       SAMPLES.  TEMP5 HAS FORMAT G/G G/G G/C . . .
TEMP←5000↑ DATA2
TEMP2←TEMP ⎕SS 'tig'
TEMP3←TEMP2/⍳⍴TEMP2
TEMP4←(TEMP3[2]-1)↑TEMP
TEMP5←(TEMP3[1]-1)↓TEMP4
INDA←(TEMP5=⎕AV[10])/⍳⍴TEMP5
TEMP7←TEMP5[3+⍳8]
ΔSCAFF←⎕FI TEMP7
SCORES←INDA[14]↓TEMP5                          ⍝  GET CALLED BASES FOR SNP
CALLEDBASE←1↑(INDA[4]↓INDA[5]↑TEMP5)

     ALLELE1←¯1↓INDA[4]↓INDA[5]↑TEMP5   ⍝  CHOOSE ALLELES FROM DATA
     ALLELE2←¯1↓INDA[5]↓INDA[6]↑TEMP5

     ALTALLELES←ALLELE1,ALLELE2

⍝ PROCESS SCORES
IND←(SCORES='/')/⍳⍴SCORES
SCORES[IND]←' '
IND←(~SCORES∈(' ',⎕AV[10]))/⍳⍴SCORES
SCORES2←SCORES[IND]
SCORES3←62 2⍴SCORES2

NONAUSTSCORES←,SCORES3[ΔNONAUSTINDEX;]      ⍝ TEST FOR VARIATION BESI
DES AUSTINII
NUCS←'ACGT*'

TEST1←+/NUCS∈NONAUSTSCORES
⍞(TEST1=1)/'  →DOWN4'    ⍝  IF NO VARIATION AFTER REMOVE AUSTINII, SK
IP SNP

TEMP8←¯1↓INDA[1]↓INDA[2]↑TEMP5
ΔSCAFFPOS←⎕FI TEMP8              ⍝  SCAFFOLD POSITION OF SNP

⍝ DETERMINE WHICH ELEMENT OF ΔLACCDS5 IS RELEVANT
IND1←ΔLACCDS5[;1]=ΔSCAFF          ⍝ BOOLEAN, 1 IF SCAFFOLD
IND2←ΔLACCDS5[;2]≤ΔSCAFFPOS       ⍝ BOOLEAN, 1 IF SNP POSITION ≥ STAR
T POSITION OF CDS FEATURE
IND3←ΔLACCDS5[;3]≥ΔSCAFFPOS       ⍝ BOOLEAN, 1 IF SNP POSITION ≤ END
POSITION OF CDS FEATURE
```

```
IND←(IND1∧IND2∧IND3)/⍳⍴IND1          ⍝ INDEX OF ENTRY THAT SATISFIES EA
CH OF ABOVE CONDITIONS
FLAG←⍴IND
→(FLAG≠0)/ DOWN20                    ⍝ IF CDS FOUND, GO TO DOWN20
⍝IF NO CDS FOUND, PROCESS SNP

NEWLINE←ΔSCAFF,¯1,ΔSCAFFPOS,¯1,¯1,¯1,¯1,¯1,¯1,ΔSNPNUM
ΔLACSNPS←ΔLACSNPS,[1](NEWLINE)       ⍝ ADD NEW LINE TO ΔLACSNPS

CODON1←'XXX'
CODON2←'XXX'
STATUS←'O'                           ⍝ THIS STATUS INDICATES NO CDS FOUND
CLINE←'    ',CODON1,' ',CODON2,' ',STATUS,7 0⍕ΔSNPNUM      ⍝ MAKE LINE
 LISTING CODONS AND STATUS (SYN VS. NON-SYN)
ΔLACCODONS←ΔLACCODONS,[1]CLINE
→DOWN4          ⍝ PROCESS NEXT SNP

DOWN20:                              ⍝ START PROCESSING SNPS FOR WHICH CDS IS F
OUND

IND4←1↑IND                           ⍝ IN CASE MORE THAN ONE ENTRY, PIC
K FIRST
LINE←,ΔLACCDS5[IND4;]                ⍝ PICK APPROPRIATE LINE FROM ΔLACCD
S5
STARTPOS←LINE[2]
END←LINE[3]
STRAND←LINE[4]
PHASE←LINE[5]

⍝ NEXT, PICK OUT CODON CORRESPONDING TO SNP

READLACSCAFF ΔSCAFF          ⍝ READ IN SCAFFOLD SEQUENCE, IN VARIABLE Δ
SEQUENCE
⍝ DIFF←ΔSCAFFPOS-STARTPOS    ⍝ DIFFERENCE BETWEEN SNP POSITION AND S
TART POSITION OF CDS FEATURE

⍎(CALLEDBASE≠ΔSEQUENCE[ΔSCAFFPOS])/'ΔUNEQUAL2←ΔUNEQUAL2+1'

→(STRAND=0)/DOWN1                    ⍝ GO TO DOWN1 IF STRAND IS NEGATIVE

⍝ CALCULATE CODON POSITIONS FOR + STRAND
STARTFRAME←STARTPOS+PHASE    ⍝ POSITION OF FIRST CODON AFTER STARTIN
G POSITION

DIFF←ΔSCAFFPOS-STARTFRAME    ⍝ NUCLEOTIDES BETWEEN STARTFRAME AND SC
AFFOLD POSITION
POSFRAME←3|DIFF              ⍝ CODON POSITION OF SNP
STCOD←ΔSCAFFPOS-POSFRAME     ⍝ START POSITION OF CODON CONTAINING TH
E SNP
CODONINDEX←STCOD,(STCOD+1),(STCOD+2)   ⍝ POSITIONS OF ALL THREE NUCS
 OF CODON
CODON←ΔSEQUENCE[CODONINDEX]   ⍝ CODON EXTRACTED FROM I. LACIDA SEQU
ENCE
CODPOS←(ΔSCAFFPOS=CODONINDEX)/⍳3   ⍝ VARIABLE POSITION IN CODON
```

```
⍝  ADD INFORMATION TO ΔLACSNPS
NEWLINE←ΔSCAFF,STARTPOS,ΔSCAFFPOS,END,STRAND,PHASE,CODONINDEX,ΔSNPNU
M
ΔLACSNPS←ΔLACSNPS,[1]NEWLINE

 TEST←',' ∈ ALLELE2      ⍝ TEST FOR WHETHER MULTIPLE ALLELES

 →(TEST=0)/DOWN2         ⍝ IF ONLY 2 ALLELES, GO TO DOWN2

 ⍝ IF > 2 ALLELES, PROCESS

 STATUS←'M'          ⍝ INDICATES SNP HAS > 2 ALLELES
 CODON1←'XXX'
CODON2←'XXX'
STATUS←'O'                      ⍝ THIS STATUS INDICATES NO CDS FOUND
CLINE←'    ',CODON1,' ',CODON2,' ',STATUS,7 0⍕ΔSNPNUM      ⍝ MAKE LINE
 LISTING CODONS AND STATUS (SYN VS. NON-SYN)
ΔLACCONDONS←ΔLACCODONS,[1]CLINE

→DOWN4                                  ⍝ PROCESS NEXT SNP

DOWN2:  ⍝ DETERMINE WHETHER VARIATION IS NON-SYNONYMOUS OR SYNONYMOU
S

⍝ MAKE ALLELES REVERSE COMPLEMENT IF STRAND IS NEGATIVE
⍎(STRAND=0)/'ALLELE1←REVCOMP ALLELE1 ⋄ ALLELE2←REVCOMP ALLELE2'

CODON1←CODON2←CODON                ⍝ INITIALIZE VARIABLES
CODON1[CODPOS]←ALTALLELES[1]     ⍝ INSERT ONE ALTERNATIVE ALLELE
CODON2[CODPOS]←ALTALLELES[2]    ⍝ INSERT OTHER ALTERNATIVE ALLELE

⍝ DETERMINE WHETHER CODONS PRODUCE SAME AA
TRANSLATE CODON1
TEMP1←AA1
TRANSLATE CODON2
TEMP2←AA1
TEST←TEMP1=TEMP2
STATUS←'N'
⍎(TEST=1)/'STATUS←''S'''
⍝ 'SYNONYMOUS VS NON-SYNONYMOUS: ',STATUS
CLINE←ALTALLELES,' ',CODON1,' ',CODON2,' ',STATUS,7 0⍕ΔSNPNUM      ⍝
MAKE LINE LISTING CODONS AND STATUS (SYN VS. NON-SYN)
ΔLACCODONS←ΔLACCODONS,[1]CLINE          ⍝ ADD LINE TO ΔLACCODONS

DOWN4:DATA2←(TEMP3[2]-1)↓DATA2        ⍝  DROP CURRENT SNP FROM DATA2

TEST←+/(DATA2 ⎕SS 'tig')

→(TEST>1)/RETI
⎕TCLF
'PROGRAM COMPLETE.  SNP DATA IN VARIABLES ''ΔLACSNPS'' AND ''ΔLACCOD
ONS''.'

****************************
```

```
SEPSNPS
⍝ THIS FUNCTION SEPARATES SNPS INTO SYNONYMOUS, NON-SYNONYMOUS AND O
THER

STATUS←ΔLACCODONS[;12]      ⍝ COLUMN OF N'S, S'S, O'S
SNPNUMS←ΔLACCODONS[;12+⍳7]    ⍝ SNP NUMBERS FROM ΔLACCODONS IN TEXT
FORMAT
BLANKS←((⍴STATUS)⍴' ')        ⍝   ADD ONE COLUMN OF SPACES
SNPNUMS←,(SNPNUMS,BLANKS)
SNPNUMS2←⎕FI SNPNUMS          ⍝ CHANGE SNP NUMBERS TO NUMERIC

⍝ GET SYN SNPS

IND1←(STATUS='S')/⍳⍴STATUS    ⍝ INDEX OF WHICH ROWS OF ΔLACODONS COR
RESPOND TO SYNONYMOUS SNPS

SYNSNPNUMS←SNPNUMS2[IND1]     ⍝ PICK SNP NUMBERS CORRESPONDING TO SY
NONYMOUS SITES

DIMS←⍴ΔSNPS1
ΔSYNSCORES←0 62 2⍴0
I←0
RETI:I←I+1

TEST←(I÷1000)=(⌈(I÷1000))
⍞(TEST)/'I'
TEST←ΔSNPS1[I]∈SYNSNPNUMS
⍞(TEST)/'ΔSYNSCORES←ΔSYNSCORES,[1]ΔSCORES[I;;]'
→(I<DIMS)/RETI

⍝ GET NON-SYN SNPS
'PROCESSING NON-SYN SNPS'

IND2←(STATUS='N')/⍳⍴STATUS
NONSNPNUMS←SNPNUMS2[IND2]

DIMN←⍴ΔSNPS1
ΔNONSCORES←0 62 2⍴0
J←0
RETJ:J←J+1

TEST←(J÷1000)=(⌈(J÷1000))
⍞(TEST)/'J'
TEST←ΔSNPS1[J]∈NONSNPNUMS
⍞(TEST)/'ΔNONSCORES←ΔNONSCORES,[1]ΔSCORES[J;;]'
→(J<DIMN)/RETJ

DOWN:'PROCESSING O SNPS'
IND3←(STATUS='O')/⍳⍴STATUS

OSNPNUMS←SNPNUMS2[IND3]

DIMO←⍴ΔSNPS1
ΔOSCORES←0 62 2⍴0
K←0
```

```
RETK:K←K+1

TEST←(K÷1000)=(⌈(K÷1000))
⍎(TEST)/'K'
TEST←ΔSNPS1[K]∈OSNPNUMS
⍎(TEST)/'ΔOSCORES←ΔOSCORES,[1]ΔSCORES[K;;]'
→(K<DIMO)/RETK
**************************

SPLICE4

⍝ THIS FUNCTION CALCULATES THE NUMBERS OF SYNONYMOUS AND NON-SYNONYM
OUS SITES INT THE TRANSCRIPTS
⍝      BASED ON MATCHES TO I. LACUNOSA CODING REGIONS

ΔTOTSYN←ΔTOTNONSYN←0
DIMTRAN←1↑⍴ΔLACTRANDATA

LACSCAFFS←ΔLACINDEX[;1]

COUNTER←1     ⍝ THIS IS A COUNTER FOR TRANSCRIPT NUMBER

ΔLACTRANIND←0⍴' '        ⍝  THIS IS THE VARIABLE THAT WILL HOLD SUCCES
SIVE  SPLICED TRANSCRIPT INDICIES

                         ⍝ FORMAT IS  > TRANSCRIPTNUMBER SCAFFNUM POSI
TIONINDEX ⎕TCLF
                         ⍝ TRANSCRIPTNUMBER IS NUMBER OF TRANSCRIPT (I
.E. 'COUNTER' IN THIS PROGRAM)
                         ⍝      SCAFFNUM IS 8 0⍕ SCAFFOLD NUMBER
                         ⍝      POSITIONINDEX IS VECTOR OF POSITIONS OF
 SPLICED TRANSCRIPT IN 10 0⍕ FORMAT

ΔLACTRANINDDATA←0 8⍴0    ⍝ THIS VARIABLE HOLDS START AND END POSITION
S OF THE TRANSCRIPT
                          ⍝ FORMAT FOR EACH ROW IS TRANSCRIPTNUMBER SC
AFFNUM, STRAND, STARTPOS, ENDPOS,
                         ⍝     START POSITION OF TRANSCRIPT DATA IN ΔL
ACTRANIND,END POSITION OF TRANSCRIPT DATA IN ΔLACTRANIND

LACTRANLENGTH←0

I←0
RETI:I←I+1
TEST←(I÷100)=(⌈(I÷100))
⍎(TEST=1)/'I'

LINE1←ΔLACTRANDATA[I;]     ⍝ READ IN DATA FOR TRANSCRIPT I
SCAFF←LINE1[2]           ⍝ LACUNOSA SCAFFOLD CORRESPONDING TO TRANSCR
IPT

⍎(~SCAFF∈LACSCAFFS)/'⎕TCLF ◇ ''NO CONTIG FOUND'' ◇ TRSH←⎕ ◇ →DOWN'

READLACSCAFF SCAFF
```

```
⍝ PICK OUT I. LACUNOSA CDS FEATURES CONTAINED IN THE TRANSCRIPT
IND1←⍋LACCDS5[;1]=SCAFF
IND2←⍋LACCDS5[;2]≥LINE1[3]     ⍝ TEST WHETHER CDS FEATURE CONTAINED I
N TRANSCRIPT
IND3←⍋LACCDS5[;3]≤LINE1[4]     ⍝ DITTO
IND←IND1∧IND2∧IND3

⍎(0=+/IND)/' →DOWN'                ⍝ SKIP IF TRANSCRIPT CONTAINS NO
I. LACCDS FEATURES

IND←IND/⍳⍴IND

PARTCDS4←⍋LACCDS5[IND;]        ⍝ PICK I. LAC CDS FEATURES CONTAINED I
N TRANSCRIPT
DIMPART←1↑⍴PARTCDS4
IND1←(PARTCDS4[;4]=1)/⍳DIMPART     ⍝ INDEX FOR CDS ON POSITIVE STRAND
IND2←(PARTCDS4[;4]=0)/⍳DIMPART     ⍝ INDEX FOR CDS ON NEGATIVE STRAND

PARTPOS←PARTCDS4[IND1;]            ⍝ PICK CDS ON POSITIVE STRAND

→(0=1↑⍴PARTPOS)/DOWN

IND←⍋PARTPOS[;2]
PARTPOS←PARTPOS[IND;]              ⍝ SORT CDS ON POSITIVE STRAND IN O
RDER OF INCREASING BEGINNING POSITION
PARTNEG←PARTCDS4[IND2;]            ⍝ PICK CDS ON NEGATIVE STRAND
IND←⍋PARTNEG[;2]
PARTNEG←PARTNEG[IND;]              ⍝ SORT CDS ON NEGATIVE STRAND IN O
RDER OF DECREASING BEGINNING POSITION

BEGPHASE←PARTPOS[1;5]

⍝ SPLICE TRANSCRIPT FOR CDS ON POSITIVE STRAND

FLAG←0                         ⍝ FLAG= 0 INDICATES THE NEXT CDS IS THE F
IRST CDS IN A GENE
SPLICEDTRAN←0⍴' '
DIM1←1↑⍴PARTPOS
→(DIM1=0)/DOWN10

⍝ START POSITIVE STRAND LOOP

CDSIND2←CDSIND←0⍴0

K←0
RETK:K←K+1
LINE2←PARTPOS[K;]              ⍝ LAC CDS DATA
PHASE←LINE2[5]

STRAND←LINE2[4]

CDSSTARTPOS←LINE2[2]
CDSENDPOS←LINE2[3]
```

```
CDSIND←CDSIND,CDSSTARTPOS,CDSENDPOS
CDSIND2←CDSIND2,((CDSSTARTPOS-1)+⍳(1+(CDSENDPOS-CDSSTARTPOS)))


DOWN1:→(K<DIM1)/RETK

     BEGPHASE←PARTPOS[1;5]
     ⍝FIRSTPOS←CDSIND[1]
       ⍎(BEGPHASE=0)/'CDSIND2←2↓CDSIND2'
       ⍎(BEGPHASE=2)/'CDSIND2←1↓CDSIND2'

    STRAND←1
    LINE5←'> ',(8 0⍕COUNTER),(8 0⍕SCAFF),(10 0⍕CDSIND),⎕TCLF

    ΔLACTRANIND←ΔLACTRANIND,LINE5

    LACTRANSTART←LACTRANLENGTH+1
    LACTRANLENGTH←LACTRANEND←LACTRANLENGTH+⍴LINE5

    LINE6←COUNTER,SCAFF,STRAND,BEGPHASE,(1↑CDSIND),(¯1↑CDSIND),LACTRA
NSTART,LACTRANEND
    ΔLACTRANINDDATA←ΔLACTRANINDDATA,[1]LINE6
    COUNTER←COUNTER+1


COUNTSYN SEQ        ⍝ CALL SCRIPT COUNTSYN
ΔTOTSYN←ΔTOTSYN+SYN
ΔTOTNONSYN←ΔTOTNONSYN+NONSYN

→(0=1↑⍴PARTNEG)/DOWN

DOWN10:

⍝ SPLICE TRANSCRIPT FOR CDS ON NEGATIVE STRAND
                   PH

FLAG←0                          ⍝ FLAG= 0 INDICATES THE NEXT CDS IS THE F
IRST CDS IN A GENE
SPLICEDTRAN←0⍴' '
DIM1←1↑⍴PARTNEG
→(DIM1=0)/DOWN
CDSIND2←CDSIND←0⍴0
J←0
RETJ:J←J+1
LINE2←PARTNEG[J;]
PHASE←LINE2[5]

CDSSTARTPOS←LINE2[2]
CDSENDPOS←LINE2[3]
INDB←(CDSSTARTPOS-1)+⍳(1+(CDSENDPOS-CDSSTARTPOS))
CDSIND←CDSIND,CDSSTARTPOS,CDSENDPOS
CDSIND2←CDSIND2,((CDSSTARTPOS-1)+⍳(1+(CDSENDPOS-CDSSTARTPOS)))
```

```
DOWN2:→(J<DIM1)/RETJ

 BEGPHASE←PARTNEG[1;5]

SPLICETRAN2←ΔSEQUENCE[CDSIND2]
SPLICETRAN←REVCOMP SPLICETRAN2
TRANSLATE SPLICETRAN

COUNT1←+/AA1='★'
COUNT2←+/AA2='★'
COUNT3←+/AA3='★'


COUNTS←COUNT1,COUNT2,COUNT3
MIN←⌊/COUNTS
INDC←(COUNTS=MIN)/⍳3
⍎((⍴INDC)>1)/'→DOWN'

CDSIND2←(¯1×(INDC-1))↓CDSIND2

STRAND←0
LINE5←'> ',(8 0⍕COUNTER),(8 0⍕SCAFF),(10 0⍕CDSIND),⎕TCLF
ΔLACTRANIND←ΔLACTRANIND,LINE5

LACTRANSTART←LACTRANLENGTH+1
LACTRANLENGTH←LACTRANEND←LACTRANLENGTH+⍴LINE5

LINE6←COUNTER,SCAFF,STRAND,BEGPHASE,(1↑CDSIND),(¯1↑CDSIND),LACTRANST
ART,LACTRANEND

ΔLACTRANINDDATA←ΔLACTRANINDDATA,[1]LINE6
COUNTER←COUNTER+1

SEQ←ΔSEQUENCE[CDSIND2]
SEQ2←REVCOMP SEQ
COUNTSYN SEQ2
ΔTOTSYN←ΔTOTSYN+SYN
ΔTOTNONSYN←ΔTOTNONSYN+NONSYN

DOWN:→(I<DIMTRAN)/RETI


*****************************
COUNTSYN  X
⍝ THIS PROGRAM CALCULATES THE NUMBER OF SYNONYMOUS AND NON-SYNONYMOU
S SITES IN A SEQUENCE X

SEQ←X
LENGTH←⍴X
MOD←3|LENGTH
SEQ←(¯1×MOD)↓SEQ

⍝  DROP LAST CODON IF IT IS STOP CODON
ENDCOD←¯3↑SEQ
```

```
TRANSLATE ENDCOD
⍎(AA1='★')/'SEQ←¯3↓SEQ'

LENGTH2←(⍴SEQ)÷3
MAT←(LENGTH2,3)⍴SEQ
MAT←⍉MAT

TEMP←+/(ΔCODONS∧.=MAT)
TEMP←((⍴TEMP),1)⍴TEMP


TEMP2←TEMP,TEMP,TEMP

TEMP3←ΔDEGENMATRIX×TEMP2

SYN←+/+/TEMP3
TOT←LENGTH2×3
NONSYN←TOT-SYN
****************************
```

Several of the above scripts call the script TRANSLATE, which translate nucleotide sequences into amino-acid sequences:

```
****************************

TRANSLATE X;MAX;CODONS;AA
MAX←⌊(⍴X)÷3
CODONS←⍉(MAX,3)⍴X
FIXFN
AA←,(1 65⍴⍳65)+.×(ΔCODE∧.=CODONS)
AA1←,ΔSYMB[AA;]
X←1↓X
MAX←⌊(⍴X)÷3
CODONS←⍉((MAX,3)⍴X)
FIXFN
AA←,(1 65⍴⍳65)+.×(ΔCODE∧.=CODONS)
AA2←,ΔSYMB[AA;]
X←1↓X
MAX←⌊(⍴X)÷3
CODONS←⍉((MAX,3)⍴X)
FIXFN
AA←,(1 65⍴⍳65)+.×(ΔCODE∧.=CODONS)
AA3←,ΔSYMB[AA;]
****************************
```

The script TRANSLATE requires the variables ΔCODE (a 65 x 3 character matrix) and ΔSYMB (a 65 x 1 character vector):

Transpose of ΔCODE =

```
TTTTTTTTTTTTTTTTCCCCCCCCCCCCCCCCAAAAAAAAAAAAAAAAGGGGGGGGGGGGGGGG-
TTTTCCCCAAAAGGGGTTTTCCCCAAAAGGGGTTTTCCCCAAAAGGGGTTTTCCCCAAAAGGGG-
TCAGTCAGTCAGTCAGTCAGTCAGTCAGTCAGTCAGTCAGTCAGTCAGTCAGTCAGTCAGTCAG-
```

Transpose of ΔSYMB =

```
FFLLSSSSYY**CC*WLLLLPPPPHHQQRRRRIIIMTTTTNNKKSSRRVVVVAAAADDEEGGGG-
```

II. Indexing the *I. lacunosa* genome. The script INDEXLAC indexes contigs in the LAC genome FASTA file. It produces the variable ΔLACINDEX, which is an N X 3 matrix. Each row of the matrix consists of the following elements:

1. scaffold number
2. start position of scaffold sequence in FASTA file
3. end position of scaffold sequence in FASTA file.

ΔLACINDEX is used primarily for retrieving contig sequences using the script READLACSCAFF (also below)

```
**********************
INDEXLAC ;STARTBYTE
⍝ THIS PROGRAM INDEXES THE SCAFFOLDS OF THE I. LACUNOSA GENOME.  IT
CREATES THE VARIABLE
⍝   ΔLACINDEX, WHICH HAS THE FOLLOWING COLUMNS:
⍝       COLUMN1     SCAFFOLD NUMBER
⍝       COLUMN2     START POSITION OF SCAFFOLD IN GENOME FASTA FILE
⍝       COLUMN3     END POSITION OF SCAFFOLD IN GENOME FASTA FILE

ΔLACINDEX←0 3⍴0

STARTBYTE←0
I←0
RETI:I←I+1

TEST←(I÷100)=(⌈(I÷100))
⍎(TEST=1)/'I'

'C:\LACGENOME' ⎕NTIE ¯1  ⍝ OPEN LAC GENOME FASTA FILE

PARTSCAFF←⎕NREAD ¯1 82 5000000 STARTBYTE
⎕NUNTIE ¯1

DIM←⍴PARTSCAFF
→(DIM=0)/0

IND←(PARTSCAFF ='>')/⍳⍴PARTSCAFF
ENDSCAFF←IND[2]-1

TEMPSCAFF←ENDSCAFF↑PARTSCAFF
IND←(TEMPSCAFF=⎕TCLF)/⍳⍴TEMPSCAFF
HEAD←IND[1]↑TEMPSCAFF
SEQ←IND[1]↓TEMPSCAFF
SEQ←(SEQ≠⎕TCLF)/SEQ
ΔSCAFF←⎕FI HEAD[17+⍳5]

ΔLACINDEX←ΔLACINDEX,[1](ΔSCAFF,(STARTBYTE),(STARTBYTE+ENDSCAFF))
STARTBYTE←STARTBYTE+ENDSCAFF

→RETI
**********************
```

```
READLACSCAFF    SCAFFN

⍝ BEFORE RUNNING THIS, MAKE SURE TO RUN 'CONVLACINDEX' TO CONVERT ΔL
ACINDEX FROM A CHARACTER
⍝     TO A NUMERIC MATRIX

⍝ THIS PROGRAM READS IN I. LAC SCAFFOLD FROM FILE C:\LACGENOME
⍝     SCAFFN IS THE SCAFFOLD NUMBER TO READ

SCAFFNUMS←ΔLACINDEX[;1]
IND←(SCAFFN=SCAFFNUMS)/⍳⍴SCAFFNUMS
TEMP1←,ΔLACINDEX[IND;]
STARTBYTE←TEMP1[2]
ENDBYTE←TEMP1[3]

'C:\LACGENOME' ⎕NTIE ¯1
 SCAFFOLD←⎕NREAD ¯1 82 (ENDBYTE-STARTBYTE) STARTBYTE
⎕NUNTIE ¯1
SCAFFNAME←24↑SCAFFOLD
ΔSEQUENCE←24↓SCAFFOLD
ΔSEQUENCE←(ΔSEQUENCE≠⎕TCLF)/ΔSEQUENCE
**********************************
```

READLACSCAFF requires running 'CONVLACINDEX' before running:

```
**********************************
CONVLACFINDEX
⍝ THIS PROGRAM CONVERTS ΔLACINDEX, A CHARACTER MATRIX, TO ΔLACINDEX,
⍝     A NUMERIC MATRIX


DIM←1↑⍴ΔLACINDEX
TEMP1←ΔLACINDEX,(DIM,3)⍴' '
TEMP2←,TEMP1
IND←(TEMP2=⎕AV[10])/⍳⍴TEMP2
TEMP2[IND]←' '
TEMP3←⎕FI TEMP2
ΔLACINDEX←(DIM,3)⍴TEMP3
**********************************
```

III. Calculating and bootstrapping π values

A. Calculate average pairwise π values for all samples; as listed below this is done for all SNPs, but the script can be modified to do just synonymous, just non-synonymous, or just non-coding SNPs.

```
**********************************

PICALC2 X;I
⍝ THIS PROGRAM CALCULATES PAIRWISE PI VALUES FOR EITHER THE FULL DAT
ASET (ALL SNPS)
⍝    OR FOR SUBSETS OF DATA (E.G. SYNONYMOUS, NON-SYNONYMOUS SITES)
⍝ IT PRODUCES A MATRIX ΔPI2 THAT HAS THE AVERAGE PAIRWISE PI VALUES
FOR EACH PAIR OF SAMPLES
⍝  PROGRAM READS IN VARIABLE ΔSCORES CREATED BY READCALLS AS X
⍝     (CAN ALSO READ IN ΔSYNSCORES OR ΔNONSCORES OR ΔRSCORES)

⍝ DIVIDER←ΔTOTSYN    ⍝  CHANGE THIS IF CALCULATING PI FOR NON-SYN SIT
ES
⍝ DIVIDER←ΔTOTNONSYN  ⍝ FOR USE WITH ΔRSCORES
DIVIDER←30036768    ⍝  TOTAL NUMBER OF SITES IN TRANSCRIPTOME

DIM←1↑⍴X

PICUM←62 62⍴0    ⍝ SET UP PI MATRIX- WILL EVENTUALLY HAVE NUMBER OF P
AIRWISE DIFFERENCES ACROSS ALL VARIABLE SNPS
PICOUNT←62 62⍴0    ⍝ SET UP MATRIX FOR CUMULATIVE COUNTS (EXCLUDES MI
SSING VALUES)

I←0
RETI:I←I+1        ⍝ SNP LOOP

TEMP6←((I÷1000)-(⌈(I÷1000)))    ⍝ REPORT SNP NUMBER EVERY 100 SNPS
⍞←(TEMP6=0)/'□TCLF ◇ I'

⍝  PICK DATA FOR NEXT SNP--STRIP OFF LEADING INFO AND LEAVE JUST NUC
LEOTIDES FOR DIFFERENT
⍝      SAMPLES.  TEMP5 HAS FORMAT G/G G/G G/C . . .

SCORES3←X[I;;]            ⍝  NOTE: MODIFY WHICH ΔSCORES TO USE (E.G.
ALL, SYNONYMOUS, ETC.)

⍝ CALCULATE PAIRWISE PI VALUES FUR CURRENT SNP
TEMP7←SCORES3∘.=SCORES3
TEMP8←+/[2]TEMP7
TEMP9←(+/[3]TEMP8)÷4
TEMP9←1-TEMP9
TIND←(SCORES3[;1]='.')/⍳62
COUNTMAT←62 62⍴1
COUNTMAT[TIND;]←0
COUNTMAT[;TIND]←0
PI←TEMP9×COUNTMAT
⍝ ADD CURRENT PAIRWISE PI VALUES TO CUMULATIVE VALUES
PICUM←PICUM+PI
```

```
PICOUNT←PICOUNT+COUNTMAT

→(I<DIM)/RETI



ΔPI2←PICUM÷DIVIDER
□TCLF
'PROGRAM PICALC2 FINISHED.'
'PAIRWISE PI VALUES IN VARIABLE ''ΔPI2''.'
**********************************
```

B. Calculate pairwise π values between and within species. This script uses ΔPI2 from PICALC2

```
**********************************
PIANAL2
⍝ THIS PROGRAM CALCULATES AVERAGE PAIRWISE DIFFERENCES (PI) FOR WITH
IN EACH SPECIES AND BETWEEN SPECIES
⍝ IT USES THE VARIABLE 'ΔPI2', WHICH IS PICUM÷ΔTOTNUC, WHERE PICUM I
S INTERMEDIATE
⍝   MATRIX FROM 'PICALC2', AND ΔTOTNUC IS TOTAL NUMBER OF SITES IN T
RANSCRPTOMES, AS CALCULATED BY 'SPLICE'

CINDEX←ΔCINDEX      ⍝ INDEX FOR WHICH CORDATATRILOBA SAMPLES BEING US
ED
LINDEX←ΔLINDEX      ⍝ INDEX FOR WHICH LACUNOSA SAMPLES BEING USED
                    ⍝ THESE INDICES CAN BE CHANGED TO LOOK AT ONLY AL
LOPATRIC OR ONLY SYMPATRIC SAMPLES


⍝ MAKE MASKS FOR PUTTING 0 ON DIAGONALS FOR WITHIN SPECIES SAMPLES
TMP←⍴CINDEX
MASKC←(TMP,TMP)⍴1
I←0
RETI:I←I+1
MASKC[I;I]←0
→(I<TMP)/RETI

TMP←⍴LINDEX
MASKL←(TMP,TMP)⍴1
J←0
RETJ:J←J+1
MASKL[J;J]←0
→(J<TMP)/⍴RETJ

⍝ CALCULATE WITHIN-CORDAT AVERAGE PI
CW1←ΔPI2[CINDEX;CINDEX]     ⍝ CHOOSE SUBSET OF ΔPI2 CORRESPONDING TO
CORDAT
CW2←CW1×MASKC               ⍝ MAKE DIAGONAL ELEMENTS 0
DIM1←⍴CINDEX                ⍝ NUMBER OF CORDAT SAMPLES
CW3←(+/+/CW2)÷(DIM1×(DIM1-1))    ⍝ AVERAGE PI FOR WITHIN CORDAT
```

```
⍝ CALCULATE WITHIN-LAC AVERAGE PI
LW1←⍙PI2[LINDEX;LINDEX]
LW2←LW1×MASKL
DIM2←⍴LINDEX
LW3←(+/+/LW2)÷(DIM2×(DIM2-1))

⍝ CALCULATE BETWEEN-SPECIES AVERAGE PI
B←⍙PI2[CINDEX;LINDEX]
B2←(+/+/B)÷(DIM1×DIM2)

⍝ COMMENT NEXT STATEMENTS IF RUNNING PIBOOT4


⎕TCLF
'AVERAGE PI WITHIN CORDAT: ',CW3
'AVERAGE PI WITHIN LAC:    ',LW3
'AVERAGE PI BETWEEN SPEC:  ',B2
**********************************
```

C. Bootstrap π values from PIANAL2 and test for species differences. By modifying indices used in third and fourth line, can test for differences using different sample sets, e.g. all CORDAT vs. all LAC, allopatric CORDAT vs. allopatric LAC, etc.

```
**********************************
PIBOOT4
⍝  THIS PROGRAM BOOTSTRAPS DIFFERENCES IN PI THE TWO SPECIES

CINDEX←⍙CINDEX     ⍝ INDEX FOR WHICH CORDAT SAMPLES TO USE
LINDEX←⍙LINDEX     ⍝ INDEX FOR WHICH LACUNOSA SAMPLES TO USE


BOOTDATA2←0 2⍴0    ⍝  VARIABLE TO HOLD BOOTSTRAP DATA

MASK4←62 62⍴1

DIM←1↑⍴⍙SCORES
II←0
RETII:II←II+1    ⍝ BOOTSTRAP LOOP
TEST←(II÷1000)=(⌈(II÷1000))
⍎(TEST)/'II'                    ⍝ PRINT II EVERY 1000 SNPS
II
⍝  CREATE BOOTSTRAP DATASET
⍝  FIRST BOOTSTRAP SAMPLES
SCORES←⍙SCORES                   ⍝ COPY SNP GENOTYPES INTO VARIABLE 'SC
ORES'
CSCORES←SCORES[;CINDEX;]         ⍝  SNP GENOTYPES FOR CORDAT
LSCORES←SCORES[;LINDEX;]         ⍝  SNP GENOTYPES FOR LAC

DIMC←⍴CINDEX
DIML←⍴LINDEX
⍝  BOOTSTRAP CORDAT AND LAC SAMPLES
RANDC←?DIMC⍴DIMC
SCORES[;CINDEX;]←SCORES[;CINDEX[RANDC];]
```

```
RANDL←?DIML⍴DIML
SCORES[;LINDEX;]←SCORES[;LINDEX[RANDL];]

⍝ NOW BOOTSTRAP SNPS
DIMSNPS←1↑⍴SCORES
RANDSNPS←?DIMSNPS⍴DIMSNPS
SCORES←SCORES[RANDSNPS;;]

PICALC2 SCORES                    ⍝ CALL PICALC2 AND PIANAL2 TO CALCULATE
PI VALUES
PIANAL2

BOOTDATA2←BOOTDATA2,[1](CW3,LW3)  ⍝ APPEND BOOTSTRAP DATA FOR PI FOR
 CORDAT AND LAC SAMPLES

→(II<1000)/RETII

⍝ CALCULATE STATISTICS
DIFF←BOOTDATA2[;1]-BOOTDATA2[;2]
PROP←+/DIFF≤0
⎕TCLF
'PROPORTION OF 1000 BOOTSTRAP SAMPLE DIFFS ≤0: ',PROP

CVALS←,BOOTDATA2[;1]
CVALS2←CVALS[⍋CVALS]
LVALS←,BOOTDATA2[;2]
LVALS2←LVALS[⍋LVALS]
CONFC←CVALS2[25,975]
CONFL←LVALS2[25,975]
⎕TCLF
'CONF INTERVAL FOR I. CORD: ',CONFC
'CONF INTERVAL FOR I. LAC:  ',CONFL
⎕TCLF
'PROGRAM PIBOOT4 COMPLETED.  DATA IN VARIABLE BOOTDATA2'
**********************************
```

D. Calculate average between-species π values for sympatric vs. allopatric comparison.

```
**********************************
PICONTRAST2
⍝ THIS PROGRAM CALCULATES PI WITHIN AND BETWEEN SYMP AND ALLOPATRIC
SAMPLES
⍝   IT USES THE ΔPI2 MATRIX CALCULATED BY 'PICALC2'

MASK4←62 62⍴1

I←0
RETI:I←I+1
MASK4[I;I]←0
→(I<62)/RETI

PI←MASK4×ΔPI2
```

```
PIBETWALLO←PI[ΔCALLO3;ΔLALLO3]
PIBETWALLOCLOSE←PI[ΔCALLOCLOSE;ΔLALLOCLOSE]
PIBETWSYMP←PI[ΔCSYMPINDEX;ΔLSYMPINDEX]
PICBETWALLOSYMP←PI[ΔCALLO3;ΔCSYMPINDEX]
PILBETWALLOSYMP←PI[ΔLALLO3;ΔLSYMPINDEX]
PICBETWALLOCLOSESYMP←PI[ΔCALLOCLOSE;ΔCSYMPINDEX]
PILBETWALLOCLOSESYMP←PI[ΔLALLOCLOSE;ΔLSYMPINDEX]


AVEBETWSYMP←(+/+/PIBETWSYMP)÷((ρΔCSYMPINDEX)×((ρΔLSYMPINDEX)))
AVEBETWALLO←(+/+/PIBETWALLO)÷((ρΔCALLO3)×((ρΔLALLO3)))
AVEBETWALLOCLOSE←(+/+/PIBETWALLOCLOSE)÷((ρΔCALLOCLOSE)×((ρΔLALLOCLOS
E)))
AVECBETWALLOSYMP←(+/+/PICBETWALLOSYMP)÷((ρΔCALLO3)×(ρΔCSYMPINDEX))
AVELBETWALLOSYMP←(+/+/PILBETWALLOSYMP)÷((ρΔLALLO3)×(ρΔLSYMPINDEX))

AVECBETWALLOCLOSESYMP←(+/+/PICBETWALLOCLOSESYMP)÷((ρΔCALLOCLOSE)×(ρΔ
CSYMPINDEX))
AVELBETWALLOCLOSESYMP←(+/+/PILBETWALLOCLOSESYMP)÷((ρΔLALLOCLOSE)×(ρΔ
LSYMPINDEX))


ΔBETWDIFF1←AVEBETWALLO-AVEBETWSYMP
ΔBETWDIFF2←AVEBETWALLOCLOSE-AVEBETWSYMP
ΔBETWDIFF3←AVECBETWALLOSYMP-AVELBETWALLOSYMP
ΔBETWDIFF4←AVECBETWALLOCLOSESYMP-AVELBETWALLOCLOSESYMP

ΔOBSPIVECTOR←AVEBETWALLO,AVEBETWALLOCLOSE,AVEBETWSYMP,ΔBETWDIFF1,ΔBE
TWDIFF2,AVECBETWALLOSYMP,AVELBETWALLOSYMP,AVECBETWALLOCLOSESYMP,AVEL
BETWALLOCLOSESYMP,ΔBETWDIFF3,ΔBETWDIFF4 ⍝ OBSERVED VALUES

⎕TCLF
'BETWEEN SPECIES COMPARISIONS'
⎕TCLF
'a. AVERAGE PI BETWEEN KNOWN ALLO: ',AVEBETWALLO
'b. AVERAGE PI BETWEEN CLOSE ALLO: ',AVEBETWALLOCLOSE
'c. AVERAGE PI BETWEEN SYMP:       ',AVEBETWSYMP
'd. DIFFERENCE a. - c.:            ',(ΔBETWDIFF1)
'e. DIFFERENCE b. - c.:            ',(ΔBETWDIFF2)
⎕TCLF
'WITHIN SPECIES COMPARISONS'
⎕TCLF
'f. I. CORD BETW ALLO(KNOWN) AND SYMP ',(AVECBETWALLOSYMP)
'g. I. LAC  BETW ALLO(KNOWN) AND SYMP ',(AVELBETWALLOSYMP)
'h. I. CORD BETW ALLO(CLOSE) AND SYMP ',(AVECBETWALLOCLOSESYMP)
'i. I. LAC  BETW ALLO(CLOSE) AND SYMP ',(AVELBETWALLOCLOSESYMP)

'j. DIFFERENCE f. - g.:               ',(ΔBETWDIFF3)
'k. DIFFERENCE h. - i.:               ',(ΔBETWDIFF4)


⎕TCLF
'PROGRAM PICONTRAST2 FINISHED'
**********************************
```

E. Bootstrap within and between species differences. There are two scripts. PIBOOT is the main script, which makes bootstrap samples then calls script PICONTRAST3.

```
**********************************
PIBOOT;K
⍝  THIS PROGRAM BOOTSTRAPS DIFFERENCES IN P BETWEEN ALLOPATRIC AND S
YMPATRIC POPULATIONS OF THE TWO SPECIES

COMBCINDEX←⍙CALLO3,⍙CSYMPINDEX       ⍝ COMBINED INDEX OF CORDAT KNOWN
 ALLOPATRIC AND SYMPATRIC SAMPLES
COMBLINDEX←⍙LALLO3,⍙LSYMPINDEX       ⍝ COMBINED INDEX OF LAC KNOWN AL
LOPATRIC AND SYMPATRIC SAMPLES

COMBCINDEX2←⍙CALLOCLOSE,⍙CSYMPINDEX   ⍝ COMBINED INDEX OF CORDAT CLO
SE ALLOPATRIC AND SYMPATRIC SAMPLES
COMBLINDEX2←⍙LALLOCLOSE,⍙LSYMPINDEX   ⍝ COMBINED INDEX OF LAC CLOSE
ALLOPATRIC AND SYMPARIC SAMPLES

CONTRDATA←0 11⍴0       ⍝ INITIALIZE MARIX TO HOLD OUTPUT DATA

MASK4←62 62⍴1

I←0
RETI:I←I+1
MASK4[I;I]←0
→(I<62)/RETI

J←0
RETJ:J←J+1     ⍝ BOOTSTRAP SAMPLE LOOP

TEMP←(J÷100)=(⌈(J÷100))
⍎(TEMP)/'J'

PITEMP←PITEMP2←⍙PI2
⍝ MAKE BOOTSTRAP SAMPLE FOR KNOWN ALLOW AND SYMP SAMPLES

CIND←?(⍴COMBCINDEX)⍴(⍴COMBCINDEX)
LIND←?(⍴COMBLINDEX)⍴(⍴COMBLINDEX)

CIND2←COMBCINDEX[CIND]
LIND2←COMBLINDEX[LIND]

PITEMP[COMBCINDEX;COMBCINDEX]←PITEMP[CIND2;CIND2]
PITEMP[COMBLINDEX;COMBLINDEX]←PITEMP[LIND2;LIND2]
PITEMP[COMBCINDEX;COMBLINDEX]←PITEMP[CIND2;LIND2]
PITEMP[COMBLINDEX;COMBCINDEX]←PITEMP[LIND2;CIND2]

⍝ MAKE BOOTSTRAP SAMPLE FOR CLOSE ALLOW AND SYMP SAMPLES

CIND←?(⍴COMBCINDEX2)⍴(⍴COMBCINDEX2)
LIND←?(⍴COMBLINDEX2)⍴(⍴COMBLINDEX2)

CIND2←COMBCINDEX2[CIND]
LIND2←COMBLINDEX2[LIND]
```

```
PITEMP2[COMBCINDEX2;COMBCINDEX2]←PITEMP2[CIND2;CIND2]
PITEMP2[COMBLINDEX2;COMBLINDEX2]←PITEMP2[LIND2;LIND2]
PITEMP2[COMBCINDEX2;COMBLINDEX2]←PITEMP2[CIND2;LIND2]
PITEMP2[COMBLINDEX2;COMBCINDEX2]←PITEMP2[LIND2;CIND2]

PICONTRAST3     ⍝ CALL SCRIPT PICONTRAST3

→(J<1000)/RETJ

⍝ CALCULATE CONFIDENCE SETS

⎕TCLF
'PROPORTION OF 1000 BOOTSTRAP SAMPLES ≥ OBSERVED PI CONTRASTS'
'   FIRST NUMBER IS OBSERVED, SECOND IS  PROPORTION'
⎕TCLF
'a. AVERAGE PI BETWEEN LAC AND CORD ALLOKNOWN POPS: ',⍕⍪OBSPIVECTOR[1
]
'b. AVERAGE PI BETWEEN LAC AND CORD ALLOCLOSE POPS: ',⍕⍪OBSPIVECTOR[2
]
'c. AVERAGE PI BETWEEN LAC AND CORD SYMP      POPS: ',⍕⍪OBSPIVECTOR[3
]
'd. DIFFERENCE a. - c.                                ',⍕⍪OBSPIVECTOR[4
],'  ',⍕+/(CONTRDATA[;4]≥⍪OBSPIVECTOR[4])÷1000
'e. DIFFERENCE b. - c.                                ',⍕⍪OBSPIVECTOR[5
],'  ',⍕+/(CONTRDATA[;5]≥⍪OBSPIVECTOR[5])÷1000

⎕TCLF
'f. AVERAGE PI BETWEEN I. CORD ALLOKNOWN AND SYMP:  ',⍕⍪OBSPIVECTOR[6
]
'g. AVERAGE PI BETWEEN I. LAC  ALLOKNOWN AND SYMP:  ',⍕⍪OBSPIVECTOR[7
]
'h. AVERAGE PI BETWEEN I. CORD ALLOCLOSE AND SYMP:  ',⍕⍪OBSPIVECTOR[8
]
'i. AVERAGE PI BETWEEN I. LAC  ALLOCLOSE AND SYMP:  ',⍕⍪OBSPIVECTOR[9
]
'j. DIFFERENCE f. - g.                                ',⍕⍪OBSPIVECTOR[1
0],'  ',⍕+/(CONTRDATA[;10]≥⍪OBSPIVECTOR[10])÷1000
'k. DIFFERENCE h. - i.                                ',⍕⍪OBSPIVECTOR[1
1],'  ',⍕+/(CONTRDATA[;11]≥⍪OBSPIVECTOR[11])÷1000

⎕TCLF
'PROGRAM PIBOOT FINISHED.'
**********************************


PICONTRAST3
⍝ THIS PROGRAM PERFORMS BOOTSTRAP ON DIFFERENCES IN PI WITHIN AND BE
TWEEN SYMP AND ALLOPATRIC SAMPLES
⍝   CALLED BY 'PIBOOT'

⍝ BEFORE RUNNING THIS PROGRAM, RUN PICONTRAST2


PI←PITEMP×MASK4
PI2←PITEMP2×MASK4
```

```
N1←⍴⍙CSYMPINDEX
N2←⍴⍙LSYMPINDEX
N3←⍴⍙CALLO3
N4←⍴⍙LALLO3
N5←⍴⍙CALLOCLOSE
N6←⍴⍙LALLOCLOSE

PIBETWALLO←PI[⍙CALLO3;⍙LALLO3]
PIBETWALLOCLOSE←PI2[⍙CALLOCLOSE;⍙LALLOCLOSE]
PIBETWSYMP←PI[⍙CSYMPINDEX;⍙LSYMPINDEX]
PIBETWSYMP2←PI2[⍙CSYMPINDEX;⍙LSYMPINDEX]     ⍝ FOR USE IN ALLOCLOSE C
OMPARISIONS

PIBETWCKNOWNCSYMP←PI[⍙CALLO3;⍙CSYMPINDEX]
PIBETWCCLOSECSYMP←PI2[⍙CALLOCLOSE;⍙CSYMPINDEX]

PIBETWLKNOWNLSYMP←PI[⍙LALLO3;⍙LSYMPINDEX]
PIBETWLCLOSELSYMP←PI2[⍙LALLOCLOSE;⍙LSYMPINDEX]

AVEBETWSYMP←(+/+/PIBETWSYMP)÷(N1×N2)   ⍝ c.
AVEBETWALLO←(+/+/PIBETWALLO)÷(N3×N4)   ⍝ a.
AVEBETWALLOCLOSE←(+/+/PIBETWALLOCLOSE)÷(N5×N6) ⍝ b.

AVEBETWCKNOWNCSYMP←(+/+/PIBETWCKNOWNCSYMP)÷(N1×N3)   ⍝ f.
AVEBETWCCLOSECSYMP←(+/+/PIBETWCCLOSECSYMP)÷(N1×N5)   ⍝ h.
AVEBETWLKNOWNLSYMP←(+/+/PIBETWLKNOWNLSYMP)÷(N2×N4)   ⍝ g.
AVEBETWLCLOSELSYMP←(+/+/PIBETWLCLOSELSYMP)÷(N2×N6)   ⍝ i.

BETW1←AVEBETWALLO-AVEBETWSYMP     ⍝ = ⍙BETWDIFF1 IN PICONTRAST2; d.
BETW2←AVEBETWALLOCLOSE-AVEBETWSYMP   ⍝ = ⍙BETWDIFF2   e.

BETWDIFF1←AVEBETWCKNOWNCSYMP-AVEBETWLKNOWNLSYMP    ⍝ = ⍙BETWDIFF3 IN
PICONTRAST2; j. = f. - g.
BETWDIFF2←AVEBETWCCLOSECSYMP-AVEBETWLCLOSELSYMP    ⍝ = k. = h. - i.

TRSH←(AVEBETWALLO, AVEBETWALLOCLOSE, AVEBETWSYMP, BETW1,BETW2, AVEBE
TWCKNOWNCSYMP,AVEBETWLKNOWNLSYMP,AVEBETWCCLOSECSYMP,AVEBETWLCLOSELSY
MP,BETWDIFF1,BETWDIFF2 )
CONTRDATA←CONTRDATA,[1]TRSH
**********************************
```

IV. Calculating and bootstrapping D values (allele frequency differences between species)

A. Calculate average D values for allopatric and symparic samples

```
**********************************
FREQCONTRAST5
⍝ THIS PROGRAM COMPARES AVERAGE PAIRWISE BETWEEN-SPECIES FREQUENCY D
IFFERENCES FOR SYMPATRIC AND KNOWN ALLOPATRIC SAMPLES
⍝   FOR EACH SNP, FREQUENCY OF ALLELE WITH HIGHEST FREQUENCY IN ALL
SAMPLES IS CALCULATED FOR 4 GROUPS:
⍝        (1) ALLOPATRIC I. CORDAT SAMPLES   (2) ALLOPATRIC I. LAC SA
MPLES   (3) SYMPATRIC I. CORDAT SAMPLES
⍝        (4) SYMPATRIC I. LAC SAMPLES
⍝   THEN THREE FREQIEMCY DIFFERENCES ARE CALCULATED: (1) |(I CORDAT
ALLO - I. SYMP ALLO)  ([2) |(I. CORDAT SYMP - I. LAC SYMP)
⍝        (3) ((2) - (1))
⍝   THESE THREE FREQUENCY DIFFERENCES ARE THEN AVERAGED OVER ALL SNP
S

ΔDIFFS←0 5⍴0
ΔDIFFS2←0 5⍴0
DIM←1↑⍴ΔSCORES
I←0
RETI:I←I+1
TEST←(I÷1000)=(⌊(I÷1000))
⍎(TEST)/'I'

SCORES←ΔSCORES[I;;]
IND←⍳61
IND2←(IND≠27)/IND
SCORES2←SCORES[IND;]
IND←(SCORES2[;1]≠'.')/⍳1↑⍴SCORES2
SCORES3←SCORES2[IND;]
COUNTS←,(+/[1]((,SCORES3)∘.=ΔALLELES))
MAX←⌈/COUNTS
IND←(COUNTS=MAX)/⍳5
MAXALLELE←'CTGA★'[IND]
MAXALLELE←1↑MAXALLELE

CASCORES←,SCORES[ΔCALLO3;]
CSSCORES←,SCORES[ΔCSYMPINDEX;]
LASCORES←,SCORES[ΔLALLO3;]
LSSCORES←,SCORES[ΔLSYMPINDEX;]

FCA←(+/CASCORES=MAXALLELE)÷(⍴CASCORES)
FCS←(+/CSSCORES=MAXALLELE)÷(⍴CSSCORES)
FLA←(+/LASCORES=MAXALLELE)÷(⍴LASCORES)
FLS←(+/LSSCORES=MAXALLELE)÷(⍴LSSCORES)
DIVIDER←¯1      ⍝ SHOULD HAVE CALLED THIS 'MULTIPLIER'
⍎(FCA>FLA)/'DIVIDER←1'

SYMPDIFF←(FCS-FLS)×DIVIDER
ALLODIFF←(FCA-FLA)×DIVIDER
CDIFF1←(FCA-FCS)×DIVIDER
```

```
LDIFF1←(FLS-FLA)×DIVIDER
DIFF←ALLODIFF-SYMPDIFF
ΔDIFFS←ΔDIFFS,[1](ALLODIFF,SYMPDIFF,DIFF,CDIFF1,LDIFF1)

ALLODIFF←FCA-FLA
→(ALLODIFF=0)/DOWN

CDIFF←(FCA-FCS)÷ALLODIFF
LDIFF←(FLS-FLA)÷ALLODIFF

ALLOMIDPOINT←(FCA+FLA)÷2
ADJALLOMID←(FCA-ALLOMIDPOINT)÷ALLODIFF

ADJSYMPMID←(LDIFF+(1-CDIFF))÷2
MIDDIFF←ADJSYMPMID-ADJALLOMID
ΔDIFFS2←ΔDIFFS2,[1](CDIFF,LDIFF,ADJALLOMID,ADJSYMPMID,MIDDIFF)

DOWN:→(I<DIM)/RETI

SUM←+/[1]ΔDIFFS
AVES←SUM÷(1↑ρΔDIFFS)
□TCLF
'AVERAGE DIFFERENCES ACROSS SNPS'
'   ALLOPATRIC DIFFERENCE: ',AVES[1]
'   SYMPATRIC DIFFERENCE:  ',AVES[2]
'   ALLO DIFF - SYMP DIFF: ',AVES[3]
' CALLO - CSYMP            ',AVES[4]
' LALLO - LSYMP            ',AVES[5]
SUM2←+/[1]ΔDIFFS2
AVES2←SUM2÷(1↑ρΔDIFFS2)
□TCLF
'AVERAGE RELATIVE ALLO-SYMP DIFF FOR C: ',AVES2[1]
'AVERAGE RELATIVE ALLO-SYMP DIFF FOR L: ',AVES2[2]
'AVERAGE ALLO MIDPOINT: ',AVES2[3]
'AVERAGE SYM MIDPOINT:  ',AVES2[4]
'ALLO - SYM MIDPOINTS:  ',AVES2[5]

FREQCONTDIST    ⍝  GENERATE DATA FOR SAS PLOT OF VARIABLES IN ΔDIFFS

□TCLF
'END PROGRAM FREQCONTRAST5'
**********************************
```

B. Bootstrapping differences. The following script calculates bootstrap 95% confidence intervals for means calculated in FREQCONTRAST5

```
**********************************
FREQCONTRAST6
⍝ THIS PROGRAM DOES BOOTSTRAPPING FOR KNOWN ALLO AND SYMP DIFFERENCE
S BETWEEN SPECIES
⍝  IT IS DERIVED FROM PROGRAM 'FREQCONTRAST'
```

```
⍝  IT COMPARES AVERAGE PAIRWISE BETWEEN-SPECIES FREQUENCY DIFFERENCE
S FOR SYMPATRIC AND ALLOPATRIC SAMPLES
⍝   FOR EACH SNP, FREQUENCY OF ALLELE WITH HIGHEST FREQUENCY IN ALL
SAMPLES IS CALCULATED FOR 4 GROUPS:
⍝         (1) ALLOPATRIC I. CORDAT SAMPLES   (2) ALLOPATRIC I. LAC SA
MPLES   (3) SYMPATRIC I. CORDAT SAMPLES
⍝         (4) SYMPATRIC I. LAC SAMPLES
⍝   THEN THREE FREQIEMCY DIFFERENCES ARE CALCULATED: (1) |(I CORDAT
ALLO - I. SYMP ALLO)  ([2) |(I. CORDAT SYMP - I. LAC SYMP)
⍝         (3) ((2) - (1))
⍝   THESE THREE FREQUENCY DIFFERENCES ARE THEN AVERAGED OVER ALL SNP
S

⍝ SHUFFLE ALLO AND SYMP SAMPLES WITHIN SPECIES

MAXJ←1000
ΔAVES←0 5⍴0

J←0
RETJ:J←J+1
'J= ',J

ΔSCORES2←ΔSCORES
IND1←?28⍴28
ΔSCORES2[;ΔCINDEX3;]←ΔSCORES2[;ΔCINDEX3[IND1];]
IND2←?28⍴28
ΔSCORES2[;ΔLINDEX3;]←ΔSCORES2[;ΔLINDEX3[IND2];]


ΔDIFFS←0 5⍴0
ΔDIFFS2←0 5⍴0
DIM←1↑⍴ΔSCORES2
I←0
RETI:I←I+1
TEST←(I÷10000)=(⌊(I÷10000))
⍎(TEST)/'I'

SCORES←ΔSCORES2[I;;]
IND←⍳61
IND2←(IND≠27)/IND
SCORES2←SCORES[IND;]
IND←(SCORES2[;1]≠'.')/⍳1↑⍴SCORES2
SCORES3←SCORES2[IND;]
COUNTS←,(+/[1]((,SCORES3)∘.=ΔALLELES))
MAX←⌈/COUNTS
IND←(COUNTS=MAX)/⍳5
MAXALLELE←'CTGA*'[IND]
MAXALLELE←1↑MAXALLELE

CASCORES←,SCORES[ΔCALLO3;]
CSSCORES←,SCORES[ΔCSYMPINDEX;]
LASCORES←,SCORES[ΔLALLO3;]
LSSCORES←,SCORES[ΔLSYMPINDEX;]

FCA←(+/CASCORES=MAXALLELE)÷(⍴CASCORES)
```

```
FCS←(+/CSSCORES=MAXALLELE)÷(ρCSSCORES)
FLA←(+/LASCORES=MAXALLELE)÷(ρLASCORES)
FLS←(+/LSSCORES=MAXALLELE)÷(ρLSSCORES)
DIVIDER←¯1
⍎(FCA>FLA)/'DIVIDER←1'

SYMPDIFF←(FCS-FLS)×DIVIDER
ALLODIFF←(FCA-FLA)×DIVIDER
CDIFF1←(FCA-FCS)×DIVIDER
LDIFF1←(FLS-FLA)×DIVIDER
DIFF←ALLODIFF-SYMPDIFF
ΔDIFFS←ΔDIFFS,[1](ALLODIFF,SYMPDIFF,DIFF,CDIFF1,LDIFF1)

DOWN:→(I<DIM)/RETI

SUM←+/[1]ΔDIFFS
AVES←SUM÷(1↑ρΔDIFFS)
ΔAVES←ΔAVES,[1]AVES
⎕TCLF
'AVERAGE DIFFERENCES ACROSS SNPS'
'   ALLOPATRIC DIFFERENCE: ',AVES[1]
'   SYMPATRIC DIFFERENCE:  ',AVES[2]
'   ALLO DIFF - SYMP DIFF: ',AVES[3]
' CALLO - CSYMP            ',AVES[4]
' LALLO - LSYMP            ',AVES[5]

→(J<MAXJ)/RETJ

⍝  REPORT CONFIDENCE INTERVALS

LOW←⌊.025×1000
UP←⌈.975×1000

Q←ΔAVES[;1]
QQ←Q[⍋Q]
⎕TCLF
'95 PERCENT CONF INTERVAL FOR ALLOPATRIC FREQ DIFF BETWEEN SPECIES:'
'          (',(6 3⍕QQ[LOW]),', ',(6 3⍕QQ[UP]),')'

Q←ΔAVES[;2]
QQ←Q[⍋Q]
⎕TCLF
'95 PERCENT CONF INTERVAL FOR SYMPATRIC FREQ DIFF BETWEEN SPECIES:'
'          (',(6 3⍕QQ[LOW]),', ',(6 3⍕QQ[UP]),')'

Q←ΔAVES[;3]
QQ←Q[⍋Q]
⎕TCLF
'95 PERCENT CONF INTERVAL FOR ALLO - SYM FREQ DIFF BETWEEN SPECIES:'
'          (',(6 3⍕QQ[LOW]),', ',(6 3⍕QQ[UP]),')'

Q←ΔAVES[;4]
QQ←Q[⍋Q]
⎕TCLF
```

```
'95 PERCENT CONF INTERVAL FOR CORD ALLO - SYMP FREQ DIFF BETWEEN SPE
CIES:'
'           (',(6 3⍕QQ[LOW]),', ',(6 3⍕QQ[UP]),')'

Q←⍙AVES[;5]
QQ←Q[⍋Q]
⎕TCLF
'95 PERCENT CONF INTERVAL FOR LAC ALLO-SYM FREQ DIFF BETWEEN SPECIES
:'
'           (',(6 3⍕QQ[LOW]),', ',(6 3⍕QQ[UP]),')'


⎕TCLF
'PROGRAM FREQCONTRAST6 COMPLETED.  DATA IN VARIABLE ⍙AVES'
**********************************
```

V. M-K Tests

A. M-K test for all cordat samples vs all lac samples. This analysis uses two scripts: RUNLARGEDIFFS calls LARGEDIFFS. The former establishes a "cutoff". If the cutoff is, say, 0.9, then it treats all SNPs with frequency differences between species >= 0.9 but less than 1 as "fixed" differences, and all SNPs with freq difference < 0.9 as polymorphisms, then performs standard M-K test. RUNLARGEDIFFS establishes different cutoffs, then calls LARGEDIFFS to perform the corresponding M-K analysis.

```
**********************************
RUNLARGEDIFFS
⍝ THIS PROGRAM RUNS 'LARGEDIFFS' FOR DIFFERENT CUTOFF VALUES

CUTOFFS←11 2⍴1 1.1 .9 1 .8 .9 .7 .8 .6 .7 .5 .6 .4 .5 .3 .4 .2 .3 .1
 .2 0 .1
⍙OUTDATA←0 5⍴0
II←0
RETII:II←II+1

CUTOFF←CUTOFFS[II;]
LARGEDIFFS

LINE←SYNFOCUS,NONFOCUS,SYNLOWER,NONLOWER,G
⍙OUTDATA←⍙OUTDATA,[1]LINE
→(II<10)/RETII

PI←⍙OUTDATA[;2]÷⍙OUTDATA[;1]
G←10 1⍴⍙OUTDATA[;5]
⎕TCLF
'PIN/PIS VALUES FOR DIFFERENT FREQUENCY-DIFFERENCE BINS'
TEMP←CUTOFFS[⍳10;],G
TEMP
**********************************



LARGEDIFFS
⍝  THIS PROGRAM IDENTIFIES SNPS WITH LARGE DIFFERENCES BETWEEN SPECI
ES FOR SCORES VARIABLE X
⍝  AND PERFORMS MK TEST FOR SELECTION


MAXI←1↑⍴⍙SYNSCORES
NUCS←'ACGT*'
⍙FIXEDSNPS←0⍴0     ⍝ THIS VARIABLE CONTAINS SNP NUMBERS OF SNPS SHOWI
NG FIXED DIFFS BETWEEN SPECIES
⍙FIXEDSNPPROPS←0 14⍴' '      ⍝  THIS IS A CHARACTER MATRIX WITH INFOR
MATION ON FIXED SNPS

CINDEX←⍙CINDEX          ⍝ USE ⍙CINDEX FOR ALL SAMPLES, ⍙CALLO3 FOR ON
LY KNOW ALLOPATRIC SAMPLES, ETC.
LINDEX←⍙LINDEX
FIXEDSYN←FIXEDNON←0
```

```
⍝ CUTOFF←1 1.05          ⍝ COMMENTED OUT WHEN RUN WITH PROGRAM RUNLARG
EDIFFS
SYNCOUNT←0  0
I←0
RETI:I←I+1
⍝ TEST←(I÷100)=(⌈(I÷100))
⍝ ⍎(TEST)/'I'
SCORES←⍙SYNSCORES[I;;]
SNPNUM←⍙SNPS1[I]

CSCORES1←,SCORES[CINDEX;]
LSCORES1←,SCORES[LINDEX;]

CSCORES←(CSCORES1 ∈'ACGT')/CSCORES1
LSCORES←(LSCORES1∈'ACGT')/LSCORES1

ALLELEIND←NUCS∈(CSCORES,LSCORES)

ALLELES←ALLELEIND/NUCS
⍎(1=⍴ALLELES)/' FIXEDSYN←FIXEDSYN+1 ⋄ →DOWN1'

ALLELE1←ALLELES[1]
ALLELE2←ALLELES[2]
FREQ1←(+/(CSCORES=ALLELE1))÷(⍴CSCORES)
FREQ2←(+/(LSCORES=ALLELE1))÷(⍴LSCORES)
DIFF←|(FREQ1-FREQ2)
⍎((DIFF≥CUTOFF[1])∧(DIFF<CUTOFF[2]))/'SYNCOUNT[1]←SYNCOUNT[1]+1'
⍎(DIFF≥CUTOFF[2])/'SYNCOUNT[2]←SYNCOUNT[2]+1'

DOWN1:→(I<MAXI)/RETI

MAXJ←1↑⍴⍙NONSCORES
NONCOUNT←0  0
J←0
RETJ:J←J+1
SCORES←⍙NONSCORES[J;;]
CSCORES←,SCORES[CINDEX;]
LSCORES←,SCORES[LINDEX;]

CSCORES←(CSCORES ∈'ACGT')/CSCORES
LSCORES←(LSCORES∈'ACGT')/LSCORES

ALLELEIND←NUCS∈(CSCORES,LSCORES)
ALLELES←ALLELEIND/NUCS

ALLELES←ALLELEIND/NUCS
⍎(1=⍴ALLELES)/'FIXEDNON←FIXEDNON+1 ⋄ →DOWN2'

ALLELE1←ALLELES[1]
ALLELE2←ALLELES[2]
FREQ1←(+/(CSCORES=ALLELE1))÷(⍴CSCORES)
FREQ2←(+/(LSCORES=ALLELE1))÷(⍴LSCORES)
DIFF←|(FREQ1-FREQ2)
⍎((DIFF≥CUTOFF[1])∧(DIFF<CUTOFF[2]))/'NONCOUNT[1]←NONCOUNT[1]+1'
⍎(DIFF≥CUTOFF[2])/'NONCOUNT[2]←NONCOUNT[2]+1'
```

```
DOWN2:→(J<MAXJ)/RETJ


SYNFOCUS←SYNCOUNT[1]
NONFOCUS←NONCOUNT[1]
SYNLOWER←(1↑⍴⍙SYNSCORES)-((+/SYNCOUNT)+FIXEDSYN)
NONLOWER←(1↑⍴⍙NONSCORES)-((+/NONCOUNT)+FIXEDNON)
⎕TCLF
'M-K TABLE FOR CUTOFF ',CUTOFF
⎕TCLF
'            SYN     NONSYN'

'FIXED    ',SYNFOCUS, NONFOCUS
'POLY     ',SYNLOWER,NONLOWER
⎕TCLF
'RATIOS   ',(SYNFOCUS÷SYNLOWER),(NONFOCUS÷NONLOWER)

ALPHA←1-(SYNFOCUS×NONLOWER)÷(NONFOCUS×SYNLOWER)
NUMBER←ALPHA×NONFOCUS
⎕TCLF
'ALPHA, NUMBER ',ALPHA,NUMBER
⎕TCLF
MATRIX←2 2⍴SYNFOCUS,NONFOCUS,SYNLOWER,NONLOWER
GTEST1 MATRIX
**********************************
```

B. M-K tests for allopatric-sympatric analysis. This analysis treats as "fixed" differences SNPs with allopatric frequency differences between CUTOFF1 and CUTOFF1 + 0.1 and sympatric frequency differences > CUTOFF2. All other SNPs are polymorphic SNPs.

```
**********************************
MKTEST1

⍝ THIS PROGRAM PERFORMS M-K TEST.  'FIXED' SAMPLES ARE THOSE FOR WHI
CH ALLOPATRIC ALLELE FREQ DIFFS
⍝   > CUTOFF1, AND SYMPATRIC ALLELE FREQ DIFFS > CUTOFF2

NUCS←'ACGT★'

CINDEX←⍙CALLO3        ⍝ USE EITHER ⍙CALLO3 OR ⍙CALLOCLOSE
LINDEX←⍙LALLO3        ⍝ DITTO

CCOMBINDEX←CINDEX,⍙CSYMPINDEX     ⍝ INDEX OF ALL C SAMPLES
LCOMBINDEX←LINDEX,⍙LSYMPINDEX     ⍝ INDEX OF ALL L SAMPLES

FIXEDSYN←FIXEDNON←0  ⍝ COUNTERS FOR NONVARIABLE SNPS AFTER '★' IS RE
MOVED

CUTOFF1←.9      ⍝ ADJUST THESE AS NEEDED
CUTOFF2←.9      ⍝ ADJUST THESE AS NEEDED

SYNCOUNT←0             ⍝ VARIABLE FOR COUNT OF SYN SNPS MEETING CUTO
FF CRITERIA
```

```
MAXI←1↑⍴∆SYNSCORES
I←0
RETI:I←I+1     ⍝  LOOP FOR SYNONYMOUS SNPS
TEST←(I÷1000)=(⌈(I÷1000))
⍎(TEST)/'''I = '',I'

SYNSCORES←∆SYNSCORES[I;;]     ⍝ PICK SCORES FOR ITH SYN SNP

CALLOSCORES←,SYNSCORES[CINDEX;]          ⍝ PICK OUT SCORES FOR CALLO
SAMPLES
LALLOSCORES←,SYNSCORES[LINDEX;]          ⍝ PICK OUT SCORES FOR LALLO
SAMPLES
CSYMPSCORES←,SYNSCORES[∆CSYMPINDEX;]      ⍝ PICK OUT SCORES FOR CSYMP
 SAMPLES
LSYMPSCORES←,SYNSCORES[∆LSYMPINDEX;]      ⍝ PICK OUT SCORES FOR LSYMP
 SAMPLES

CALLOSCORES←(CALLOSCORES∈'ACGT')/CALLOSCORES   ⍝ REMOVE SCORES THAT A
RE '*' OR '.'
LALLOSCORES←(LALLOSCORES∈'ACGT')/LALLOSCORES   ⍝ DITTO
CSYMPSCORES←(CSYMPSCORES∈'ACGT')/CSYMPSCORES
LSYMPSCORES←(LSYMPSCORES∈'ACGT')/LSYMPSCORES

ALLSCORES←CALLOSCORES,LALLOSCORES,CSYMPSCORES,LSYMPSCORES
ALLELEIND←'ACGT'∈(ALLSCORES)

⍎(1=+/ALLELEIND)/' FIXEDSYN←FIXEDSYN+1 ⋄ →DOWN1'    ⍝  IF ONLY 1 ALLE
LE, SKIP

NUMA←+/ALLSCORES='A'            ⍝ COUNT NUMBERS OF EACH NUCLEOTIDE IN
 TOTAL SAMPLE
NUMC←+/ALLSCORES='C'
NUMG←+/ALLSCORES='G'
NUMT←+/ALLSCORES='T'
NUMS←NUMA,NUMC,NUMG,NUMT
MAX←⌈/NUMS                      ⍝ PICK OUT ALLELE WITH LARGEST COUNT
IND←1↑(NUMS=MAX)/⍳4
ALLELE1←'ACGT'[IND]

FREQ1←(+/(CALLOSCORES=ALLELE1))÷(⍴CALLOSCORES)     ⍝ CALC ALLELE FREQ
UENCIES IN DIFFERENT SAMPLES
FREQ2←(+/(LALLOSCORES=ALLELE1))÷(⍴LALLOSCORES)
FREQ3←(+/(CSYMPSCORES=ALLELE1))÷(⍴CSYMPSCORES)
FREQ4←(+/(LSYMPSCORES=ALLELE1))÷(⍴LSYMPSCORES)

DIFF1←|(FREQ1-FREQ2)       ⍝  CALCULATE ABSOLUATE VALUE OF DIFFERENCE
 IN ALLOPATRIC FREQUENCIES
DIFF2←FREQ3-FREQ4          ⍝  CALCULATE DIFFERENCE IN SYMPATRIC FREQU
ENCIES
⍎((FREQ1-FREQ2)<0)/'DIFF2←¯1×DIFF2'      ⍝ ADJUST DIFF IN SYMP FREQS
TO CORRECT FOR WHICH SPECIES HAS LARGER FREQ

⍎((DIFF1≥CUTOFF1)∧(DIFF2≥CUTOFF2))/'SYNCOUNT←SYNCOUNT+1'    ⍝ ADD 1 T
O SYNCOUNT IF CRITERION MET
```

```
DOWN1:→(I<MAXI)/RETI

NONCOUNT←0              ⍝ VARIABLE FOR COUNT OF NONSYN SNPS MEETING CU
TOFF CRITERIA
MAXJ←1↑⍴⍙NONSCORES
J←0
RETJ:J←J+1     ⍝  LOOP FOR NONSYNONYMOUS SNPS
TEST←(J÷1000)=(⌈(J÷1000))
⍎(TEST)/'''J = '',J'

NONSCORES←⍙NONSCORES[J;;]     ⍝ PICK SCORES FOR ITH SYN SNP

CALLOSCORES←,NONSCORES[CINDEX;]          ⍝ PICK OUT SCORES FOR CALLO
SAMPLES
LALLOSCORES←,NONSCORES[LINDEX;]          ⍝ PICK OUT SCORES FOR LALLO
SAMPLES
CSYMPSCORES←,NONSCORES[⍙CSYMPINDEX;]      ⍝ PICK OUT SCORES FOR CSYMP
 SAMPLES
LSYMPSCORES←,NONSCORES[⍙LSYMPINDEX;]      ⍝ PICK OUT SCORES FOR LSYMP
 SAMPLES

CALLOSCORES←(CALLOSCORES∈'ACGT')/CALLOSCORES  ⍝ REMOVE SCORES THAT A
RE '*' OR '.'
LALLOSCORES←(LALLOSCORES∈'ACGT')/LALLOSCORES  ⍝ DITTO
CSYMPSCORES←(CSYMPSCORES∈'ACGT')/CSYMPSCORES
LSYMPSCORES←(LSYMPSCORES∈'ACGT')/LSYMPSCORES

ALLSCORES←CALLOSCORES,LALLOSCORES,CSYMPSCORES,LSYMPSCORES
ALLELEIND←'ACGT'∈(ALLSCORES)
⍎(1=+/ALLELEIND)/' FIXEDNON←FIXEDNON+1 ⋄ →DOWN2'    ⍝  IF ONLY 1 ALLE
LE, SKIP

NUMA←+/ALLSCORES='A'            ⍝ COUNT NUMBERS OF EACH NUCLEOTIDE IN
 TOTAL SAMPLE
NUMC←+/ALLSCORES='C'
NUMG←+/ALLSCORES='G'
NUMT←+/ALLSCORES='T'
NUMS←NUMA,NUMC,NUMG,NUMT
MAX←⌈/NUMS                      ⍝ PICK OUT ALLELE WITH LARGEST COUNT
IND←1↑(NUMS=MAX)/⍳4
ALLELE1←'ACGT'[IND]

FREQ1←(+/(CALLOSCORES=ALLELE1))÷(⍴CALLOSCORES)     ⍝ CALC ALLELE FREQ
UENCIES IN DIFFERENT SAMPLES
FREQ2←(+/(LALLOSCORES=ALLELE1))÷(⍴LALLOSCORES)
FREQ3←(+/(CSYMPSCORES=ALLELE1))÷(⍴CSYMPSCORES)
FREQ4←(+/(LSYMPSCORES=ALLELE1))÷(⍴LSYMPSCORES)

DIFF1←|(FREQ1-FREQ2)       ⍝  CALCULATE ABSOLUATE VALUE OF DIFFERENCE
 IN ALLOPATRIC FREQUENCIES
DIFF2←FREQ3-FREQ4          ⍝  CALCULATE DIFFERENCE IN SYMPATRIC FREQU
ENCIES
⍎((FREQ1-FREQ2)<0)/'DIFF2←¯1×DIFF2'      ⍝ ADJUST DIFF IN SYMP FREQS
TO CORRECT FOR WHICH SPECIES HAS LARGER FREQ
```

```
⍎((DIFF1≥CUTOFF1)∧(DIFF2≥CUTOFF2))/'NONCOUNT←NONCOUNT+1'    ⍝ ADD 1 T
O NONNCOUNT IF CRITERION MET

DOWN2:→(J<MAXJ)/RETJ

SYNFOCUS←SYNCOUNT
NONFOCUS←NONCOUNT
SYNLOWER←(1↑⍴△SYNSCORES)-(SYNCOUNT+FIXEDSYN)
NONLOWER←(1↑⍴△NONSCORES)-(NONCOUNT+FIXEDNON)
⎕TCLF
'M-K TABLE FOR CUTOFFS 1 AND 2 ',CUTOFF1,CUTOFF2
⎕TCLF
'            SYN     NONSYN'

'FIXED    ',SYNFOCUS, NONFOCUS
'POLY     ',SYNLOWER,NONLOWER
⎕TCLF
'RATIOS   ',(SYNFOCUS÷SYNLOWER),(NONFOCUS÷NONLOWER)

ALPHA←1-(SYNFOCUS×NONLOWER)÷(NONFOCUS×SYNLOWER)
NUMBER←ALPHA×NONFOCUS
⎕TCLF
'ALPHA, NUMBER ',ALPHA,NUMBER
⎕TCLF
MATRIX←2 2⍴SYNFOCUS,NONFOCUS,SYNLOWER,NONLOWER
GTEST1 MATRIX
⎕TCLF
'END PROGRAM MKTEST1'
**********************************
```

C. Messer-Petrov analysis of α.

1. This analysis requires identification of ancestral allele at each SNP. To do this, the scripts 'PICKTRIF' and 'PICKTRILO' were used to identify alleles in *I. trifida* and *I. triloba* corresponding to the previously identified SNPs in the transcriptome. This was done using maf files from the alignment of the *I. lacunosa* genome to the *I. trifida* and *I. triloba* genomes. Below is the listing of 'PICKTRIF'. The script 'PICKTRILO' is identical except for it referral to the *I. triloba* genome. The programs produce the vectors 'TRIFNUCS' and 'TRILONUCS', which contain ancestral nucleotides in position corresponding to appropriate SNP (e.g. position corresponding to positions in ΔTIGS and ΔPOS.

```
**********************************
PICKTRIF
⍝  THIS PROGRAM READS IN LAC ALIGNMENT TO TRIFIDA FROM C:\TRIFLAC.TX
T AND DETERMINS THE TRIF NUCLEOTIDE
⍝   CORRESPONDING TO LAC SNPS.

STARBYTE←0
LOWER←'actg'
```

```
UPPER←'ACTG'

BADCONTIG← 0 3⍴0
DIM←1↑⍴⍙SCORES

TRIFNUCS←DIM⍴' '     ⍝  THIS WILL HOLD TRIFIDA NUCLEOTIDES; ORDER COR
RESPONDS TO ORDER IN ⍙SCORES
TYPESNP←DIM⍴' '      ⍝  THIS WILL HOLD TYPE OF SNP (E.G. SYN, NONSYN,
 OTHER)

CURRCONTIG←0

I←0

⍝  READ IN FIRST PART OF FILE

'C:\TRIFLAC.TXT' ⎕NTIE ¯1
BUFFER←⎕NREAD (¯1,82,3000000,STARBYTE)      ⍝  READ IN 3M CHARACTERS
⎕NUNTIE ¯1

IND11←(BUFFER=⎕AV[11])/⍳⍴BUFFER             ⍝  MAKE AN INDEX OF WHERE
⎕AV[11]'S ARE
HEADER←IND11[4]↑BUFFER                      ⍝  HEADER
BUFFER←IND11[4]↓BUFFER                      ⍝  STRIP HEADER FROM BUFFE
R

UP2:

READBUFF  ⍝ CALL 'READBUFF' TO PICK FIRST SEQUENCE SEGMENT FROM BUFF
ER

DOWN6: ⍝  SET INFO TO 'OLD' INFO
OLDSEQUENCE←SEQUENCE
OLDSOURCESIZE←SOURCESIZE
OLDSTRAND←STRAND
OLDSIZE←SIZE
OLDSTART←START
OLDSPECIES←SPECIES
OLDCONTIG←CONTIG
OLDKEEPINFO←KEEPINFO

UP1:  READBUFF  ⍝ CALL 'READBUFF' TO PICK NEXT SEQUENCE SEGMENT FROM
 BUFFER

I←I+1

→(SPECIES='T')/DOWN1  ⍝  SKIP TO DOWN 1 IF READ SEQUENCE IS FROM TRI
FIDA
⍝  IF ANOTHER LAC SEQUENCE SET INFO TO 'OLD' INFO
OLDSEQUENCE←SEQUENCE
OLDSOURCESIZE←SOURCESIZE
OLDSTRAND←STRAND
OLDSIZE←SIZE
OLDSTART←START
OLDSPECIES←SPECIES
```

```
OLDKEEPINFO←KEEPINFO

→UP1     ⍝ GO UP AND READ IN NEXT SEQUENCE AND INFO

DOWN1:

⍝ PICK OUT TRIF NUCS

IND1←∆TIGS=OLDCONTIG                    ⍝ INDEX FOR ALL SNPS WITH CURRE
NT CONTIG
OLDEND←(¯1+OLDSTART+OLDSIZE)            ⍝ POSITION OF END OF SEQUENCE O
N CONTIG
IND2←(∆POS≥OLDSTART)∧(∆POS≤OLDEND)      ⍝ INDEX FOR ALL SNPS WITH POSIT
ION BETWEEN BEGINNING AND END OF SEQUENCE
IND3←IND1∧IND2                          ⍝ LOGICAL AND FOR IND1 AND IND1
: ALL SNPS WITHIN SEQUENCE
IND3←IND2∧IND3
NUMS←(IND3)/⍳⍴IND3                      ⍝ SNP NUMBER FOR THOSE SNPS (I.
E. POSITION IN ∆SCORES)

→(0=⍴NUMS)/DOWN2                        ⍝ SKIP SEQUENCE IF NO SNPS
⍝ PROCESS SNPS
READLACSCAFF OLDCONTIG                  ⍝ CALL READLACSCAFF TO READ IN
LAC CONTIG SEQUENCE

IND4←(OLDSEQUENCE≠'-')                    ⍝ INDEX OF '-'S
LACSEQ←IND4/OLDSEQUENCE                 ⍝ COMPRESS -S OUT OF LAC SEQUEN
CE
TRIFSEQ←IND4/SEQUENCE                   ⍝ COMPRESS CORRESPONDING POSITI
ONS OUT OF TRIF SEQUENCE

DIM2←⍴NUMS
K←0
RETK:K←K+1
⎕TCLF
'**************************************************************************
************************'

⎕TCLF
'PROCESSING SNP'

CURNUM←NUMS[K]
CURPOS←∆POS[CURNUM]

LACSEQ2←(OLDSTART⍴'-'),LACSEQ
TRIFSEQ2←(OLDSTART⍴'-'),TRIFSEQ


UP3: ⎕TCLF
'LAC, TRIF, AND GENOME SEQS'

SEQ3←OLDSTART↓∆SEQUENCE
SIZE2←⌊/(OLDSIZE,200)
LACSEQ[⍳SIZE2]
TRIFSEQ[⍳SIZE2]
```

```
SEQ3[⍳SIZE2]

SC←,⍙SCORES[CURNUM;;]
LACSNP←LACSEQ2[CURPOS]
TRIFSNP←TRIFSEQ2[CURPOS]
GENSNP←⍙SEQUENCE[CURPOS]

⎕TCLF
'CONTIG ',OLDCONTIG,'  PROCESSING ',(1+(STARBYTE÷3000000)),'TH 3MB S
EGMENT'
BPPROC←(3000000-⍴BUFFER)
'     ',BPPROC,' BASES OF SEGMENT PROCESSED'
⎕TCLF
'SNP SCORES'
SC
⎕TCLF
'LAC, TRIF, AND GENOME NUCLEOTIDE'
LACSNP
TRIFSNP
GENSNP

IND21←(TRIFSEQ≠'-')
LACSEQ3←IND21/LACSEQ
TRIFSEQ3←IND21/TRIFSEQ

PCTEQ←(+/LACSEQ3=TRIFSEQ3)÷(⍴LACSEQ3)
'PCTEQ ',PCTEQ
⍎(PCTEQ<.5)/'TRIFNUCS[CURNUM]←''M'' ⋄ ''M INSERTED'' ⋄ →DOWN4'
       ⍝ INSERTS M INTO TRIFNUCS TO SIGNIFY UNRELIABLE ALIGNMENT

TEST3←LACSNP∈SC
'TEST3  ',TEST3
⍎(TEST3=0)/'TRIFNUCS[CURNUM]←''N'' ⋄ ''N INSERTED''⋄ →DOWN4'
     ⍝  INSERTS N INTO TRIFNUCS TO SIGNIFY MISMATCH BETW ALLELES AND
 LACSNP

SEQ4←SEQ3[⍳OLDSIZE]
PCTEQ2←(+/LACSEQ=SEQ4)÷(⍴LACSEQ)
'PCTEQ2 ',PCTEQ2
LINEINFO←(CURNUM,CURPOS,OLDCONTIG)

⍎(PCTEQ2<0.9)/'BADCONTIG←BADCONTIG,[1]LINEINFO ⋄ ''BADCONTIG'' '
                                                ⍝  INDICATES MISAL
IGNEMENT BETW LAC SEQ AND GENOME SCAFFOLD
TRIFNUCS[CURNUM]←TRIFSNP

⎕TCLF
'INSERTED NUCLEOTIDE: ',TRIFSNP

DOWN4:TYPESNP[CURNUM]←TYPE

TRSH←⎕DL 2
→(K<DIM2)/RETK

DOWN2:  ⍝ 'BUFFER BEFORE RETURN'
```

```
DOWN4:→(100000<ρBUFFER)/DOWN5
STARBYTE←STARBYTE+3000000
'C:\TRIFLAC.TXT' □NTIE ¯1
BUFFER←BUFFER,□NREAD (¯1,82,3000000,STARBYTE)      ⍝  READ IN 3M MORE
 CHARACTERS
□NUNTIE ¯1

□TCLF
'3 M MORE BYTES ADDED TO ''BUFFER''. ',(STARBYTE÷3000000),'TH SEGMEN
T READ.'

DOWN5:→UP2

□TCLF
'PROGRAM TRIFNUCS FINISHED.  DATA IN VARIABLE ''TRIFNUCS''.'
**********************************
```

The following scripts are called by 'PICTRIF' and 'PICKTRILO':

a. READBUFF

```
**********************************
READBUFF

⍝  THIS PROGRAM CALLED BY PICKTRIF
⍝   IT READS IN NEXT ALIGNMENT FROM VARIABLE 'BUFFER'

UP1:

IND11←2↑((BUFFER=□AV[11])/⍳ρBUFFER)                    ⍝ POSITION OF F
IRST □AV[11] IN BUFFER
INFO←¯1↓(IND11[1]↑BUFFER)                          ⍝ GET INFO FOR NEXT ALIG
NMENT
KEEPINFO←INFO

BUFFER←IND11[1]↓BUFFER                              ⍝ DROP INFO FROM BUFFER

IND10←(INFO=□AV[10])/⍳ρINFO                        ⍝  MAKE INDEX OF WHERE □AV
[10]'S ARE IN INFO
REST←IND10[6]↑INFO                                 ⍝  REST OF INFO BESIDES SE
QUENCE

SEQUENCE←(IND10[6]↓INFO)                       ⍝  PUT SEQUENCE IN 'SEQUENCE'
SEQUENCE←(UPPER,□AV)[(LOWER,□AV)⍳SEQUENCE]
END←¯1↑SEQUENCE
⍎(END∈(□AV[10 11]))/'SEQUENCE←¯1↓SEQUENCE'

SOURCESIZE←□FI ¯1↓(IND10[5]↓REST )                  ⍝ SIZE OF ENTIRE SOURC
E SEQUENCE, NOT JUST PARTS INVOLVED IN ALIGNMENT
REST←IND10[5]↑REST                                 ⍝ REST OF INFO BESIDES SOU
RCESIZE

STRAND←1↑(IND10[4]↓REST)                           ⍝  STRAND (+ OR -)
```

```
REST←IND10[4]↑REST                          ⍝ REST OF INFO BESIDES ST
RAND

SIZE←⌷FI ¯1↓(IND10[3]↓REST )                ⍝ SIZE OF ALIGNING REGION
 IN LAC
REST←IND10[3]↑REST                          ⍝ REST OF INFO BESIDES SI
ZE

                                            ⍝ EQUAL TO NUMBER OF NON-
DASH CHARACTERS

START←⌷FI ¯1↓(IND10[2]↓REST)                ⍝ START POSITION OF ALIGN
ING REGION ON LAC CONTIG
REST←IND10[2]↑REST                          ⍝ REST OF INFO BESIDES STA
RT

TEMP←IND10[1]↓REST                          ⍝ INFO ON CONTIG NUMBER
CONTIG←⌷FI (TEMP∈'0123456789')/TEMP         ⍝ CONTIG NUMBER

TEST←+/(TEMP ⌷SS 'lac')
⍎(TEST=1)/'SPECIES←''L'''

TEST←+/(TEMP ⌷SS 'trif')
⍎(TEST=1)/'SPECIES←''T'''

⍎(BUFFER[1]='a')/'BUFFER←2↓BUFFER'

⍎(BUFFER[2]='a')/'BUFFER←3↓BUFFER'
**********************************
```

b. READLACSCAFF. This program reads in the sequence of a particular lacunosa genome contig (SCAFFN) into variable ΔSEQUENCE

```
**********************************
READLACSCAFF SCAFFN

⍝ BEFORE RUNNING THIS, MAKE SURE TO RUN 'CONVLACINDEX' TO CONVERT ΔL
ACINDEX FROM A CHARACTER
⍝     TO A NUMERIC MATRIX

⍝ THIS PROGRAM READS IN I. LAC SCAFFOLD FROM FILE C:\LACGENOME
⍝     SCAFFN IS THE SCAFFOLD NUMBER TO READ



SCAFFNUMS←ΔLACINDEX[;1]
IND←(SCAFFN=SCAFFNUMS)/⍳⍴SCAFFNUMS
TEMP1←,ΔLACINDEX[IND;]
STARTBYTE←TEMP1[2]
ENDBYTE←TEMP1[3]


'C:\LACGENOME' ⎕NTIE ¯1
 SCAFFOLD←⎕NREAD ¯1 82 (ENDBYTE-STARTBYTE) STARTBYTE
⎕NUNTIE ¯1
SCAFFNAME←24↑SCAFFOLD
ΔSEQUENCE←24↓SCAFFOLD
ΔSEQUENCE←(ΔSEQUENCE≠⎕TCLF)/ΔSEQUENCE
**********************************
```

2. Ancestral alleles in TRIFNUCS and TRILONUCS are merged with script 'MERGE'. Ancestral alleles from TRILONUCS take precedence over those from TRIFNUCS when both are identified for a given SNP.

```
**********************************
X MERGE Y
⍝ THIS PROGRAM MERGES RESULTS FROM TRIFIDA AND TRILOBA
⍝  X IS TRIFNUCS, Y IS TRILONUCS

MAXI←⍴Y

MERGED←0⍴' '

I←0
RETI:I←I+1

X1←X[I]
Y1←Y[I]

⍎(X1∈'ACGT')/'MERGED←MERGED,X1 ⋄  →DOWN'
⍎(Y1∈'ACGT')/'MERGED←MERGED,Y1 ⋄ →DOWN'
```

```
MERGED←MERGED,' '

DOWN:→(I<MAXI)/RETI

⎕TCLF
'TOTAL SNPS IDENTIFIED = ',+/MERGED∈'ACGT'
⎕TCLF

'PROGRAM MERGE FINISHED.  DATA IN VARIABLE ''MERGED''.'
**********************************
```

3. Ancestral nucleotides are separated for synonymous, non-synonymous, and non-coding SNPs, and then for each SNP pair the Messer-Petrov α is calculated, as well as distance separating the two SNPs. The script produces a matrix 'MATRIX' that has 5 columns:

Col1: midpoint of distance bin (distance between SNPS)
Col2: α(d) for each distance bin for fixed differences (non-syn vs. syn SNPs)
Col3: α(d) for each distance bin for nearly fixed differences (non-syn vs. syn SNPs)
Col4: α(d) for each distance bin for fixed differences (non-coding vs. syn SNPs)
Col5: α(d) for each distance bin for nearly fixed differences (non-coding vs. syn SNPs)

It also produces vectors A1A, A1B, A2A and A2B corresponding to cols 2 – 5.
The matrix is used for analysis in SAS. The vectors are used for analysis in Mathematica

```
**********************************
SEPMERGEDNUCS

⍝  THIS PROGRAM SEPARATES MERGED NUCS INTO SYN, NON-SYN, AND REG ANC
ESTRAL NUCS
⍝     THEN ANALYZES THEM FOR MESSER AND PETROV MK TESTS
⍝  USES ONLY ALL  SAMPLES

MAXI←⍴⍙MERGED     ⍝ ⍙MERGED IS SAME AS 'MERGED' PRODUCED BY PROGRAM '
MERGE'

NUMS←⎕FI ,⍙LACCODONS[;13+⍳6]    ⍝ SNP NUMBERS FROM ⍙LACCODONS

CSYNFREQS←LSYNFREQS←CNONFREQS←LNONFREQS←CRFREQS←LRFREQS←0⍴0    ⍝  HOL
D COUNTS

I←0
RETI:I←I+1

SC←⍙SCORES[I;;]            ⍝ READ IN SCORES FOR SNP I
CSC←,SC[⍙CINDEX;]          ⍝ PICK OUT CORD SCORES
CSC←(CSC∈'ACGT')/CSC       ⍝ GET RID OF '.'
LSC←,SC[⍙LINDEX;]          ⍝ PICK OUT LAC SCORES
LSC←(LSC∈'ACGT')/LSC       ⍝ GET RID OF '.'
BOTH←CSC,LSC               ⍝ COMBINE CORD AND LAC SCORES

TNUC←⍙MERGED[I]           ⍝ PICK CORRESPONDING MERGED NUC
→(TNUC∈' NM')/DOWN         ⍝  IF NUC∈NM, SKIP
```

```
SNP←ΔLACSNPS[I;10]          ⍝ READ SNP NUMBER FROM ΔLACSNPS
TEST←SNP∈NUMS               ⍝ TEST IF SNP NUMBER IN ΔLACCODONS
→(TEST=0)/DOWN              ⍝ IF NOT, SKIP

IND←(NUMS=SNP)/⍳⍴NUMS       ⍝ POSITION OF SNP IN ΔLACCODONS
TYPE←ΔLACCODONS[IND;12]     ⍝ GET TYPE (NON-SYN, SYN, ETC.

→(TYPE='O')/DOWN            ⍝ SKIP IF TYPE='O'

TEST←(TNUC∈BOTH)            ⍝ TEST IF MERGED ALLELE IN LAC OR CORD (AN
CESTRAL ALLELE)
→(TEST=0)/DOWN              ⍝ SKIP IF NOT


CFREQ←(+/CSC=TNUC)÷⍴CSC     ⍝ ANCESTRAL ALLELE FREQ IN CORD
LFREQ←(+/LSC=TNUC)÷⍴LSC     ⍝ ANCESTRAL ALLELE FREQ IN LAC

CFREQ←1-CFREQ               ⍝ FREQ OF NEW ALLELE IN CORD
LFREQ←1-LFREQ               ⍝ FREQ OF NEW ALLELE IN LAC

⍝ APPEND NEW ALLELE FREQ TO APPROPRIATE VECTOR

⍎(TYPE='N')/'CNONFREQS←CNONFREQS,CFREQ ⋄ LNONFREQS←LNONFREQS,LFREQ'
⍎(TYPE='S')/'CSYNFREQS←CSYNFREQS,CFREQ ⋄ LSYNFREQS←LSYNFREQS,LFREQ'
⍎(TYPE='R')/'CRFREQS←CRFREQS,CFREQ ⋄ LRFREQS←LRFREQS,LFREQ'


DOWN:→(I<MAXI)/RETI

⍝ BIN THE FREQUENCIES

NUMBINS←50     ⍝ CHANGE THIS TO WHATEVER
BINS←(⍳NUMBINS)÷NUMBINS
BINS←BINS-BINS[1]

NBINS←+/CNONFREQS∘.≥BINS
CNONTOT←+/[1]NBINS∘.=(⍳NUMBINS)

NBINS←+/LNONFREQS∘.≥BINS
LNONTOT←+/[1]NBINS∘.=(⍳NUMBINS)

NBINS←+/CSYNFREQS∘.≥BINS
CSYNTOT←+/[1]NBINS∘.=(⍳NUMBINS)

NBINS←+/LSYNFREQS∘.≥BINS
LSYNTOT←+/[1]NBINS∘.=(⍳NUMBINS)

NBINS←+/CRFREQS∘.≥BINS
CRTOT←+/[1]NBINS∘.=(⍳NUMBINS)

NBINS←+/LRFREQS∘.≥BINS
LRTOT←+/[1]NBINS∘.=(⍳NUMBINS)

NONTOT←CNONTOT+LNONTOT      ⍝ SUM NON-SYN SNPS OVER SPECIES
```

```
SYNTOT←CSYNTOT+LSYNTOT          ⍝ SUM SYN SNPS OVER SPECIES

RTOT←CRTOT+LRTOT                ⍝ SUM NON-CODING SNPS OVER SPECIES

⍝  CALCULATE ALPHA VECTORS FOR MESSER AND PETROV MK
⍝  NON-SYN FREQ DIFF = 1
DS←169
DN←190
ALPHA1A←1 - (DS÷DN)×NONTOT÷SYNTOT

⍝ NON-SYN FREQ DIFF [0.9, 1)
DS←699
DN←644
ALPHA1B←1 - (DS÷DN)×(NONTOT÷SYNTOT)

⍝ NON-CODING FREQ DIFF=1
DR←107
DS←169
ALPHA2A←1 - (DS÷DR)×(RTOT÷SYNTOT)

⍝ NON-CODING FREQ DIFF [0.9, 1)
DS←699
DR←333
ALPHA2B←1 - (DS÷DR)×(RTOT÷SYNTOT)

⍝ CONVERT DATA TO MATRIX FOR SAS
MIDPOINTS←((⍳NUMBINS)÷NUMBINS)-(1÷(2×NUMBINS))  ⍝ MIDPOINTS OF BINS

COUNT←+/BINS<.9      ⍝ NUMBER OF BINS WITH FREQ ≤ .9

MATRIX←(COUNT 1⍴MIDPOINTS),(COUNT 1⍴ALPHA1A),(COUNT 1⍴ALPHA1B),(COUN
T 1⍴ALPHA2A),(COUNT 1⍴ALPHA2B)

⍝ CONVERT DATA TO MATRICES FOR MATHEMATICA

TMP←(COUNT 1⍴MIDPOINTS),(COUNT 1⍴ALPHA1A)
CONVMATH2 TMP
A1A←CONVDATA

TMP←(COUNT 1⍴MIDPOINTS),(COUNT 1⍴ALPHA1B)
CONVMATH2 TMP
A1B←CONVDATA

TMP←(COUNT 1⍴MIDPOINTS),(COUNT 1⍴ALPHA2A)
CONVMATH2 TMP
A2A←CONVDATA

TMP←(COUNT 1⍴MIDPOINTS),(COUNT 1⍴ALPHA2B)
CONVMATH2 TMP
A2B←CONVDATA

SKIP:

⎕TCLF
```

```
'PROGRAM SEPMERGEDNUCS2 FINISHED. DATA FOR SAS IN ''MATRIX''.  DATA
FOR MATHEMATICA IN A1A, A1B,A2A,A2B.'
**********************************
```

4. Estimating α(1) (asymptotic value of α) using SAS. The data in 'MATRIX' produced by SEPMERGEDNUCS (immediately above) is cut and pasted into the following SAS program to estimate non-linear regression coefficients. This example is for an analysis of non-synonymous vs synonymous SNPs from all samples, and fixed differences between species.

```
**********************************
data alpha1;
  input x a1a a1b a2a a2b;
  cards;
[data MATRIX]
   run;

proc nlin data=alpha1;
   parms a=.042 b=.937 c=.8353;
   model a1a=1-(a+b*exp(-c*x));
   title nonlin fit for all samples, freq diff = 1, non-syn;
   run;

**********************************
```

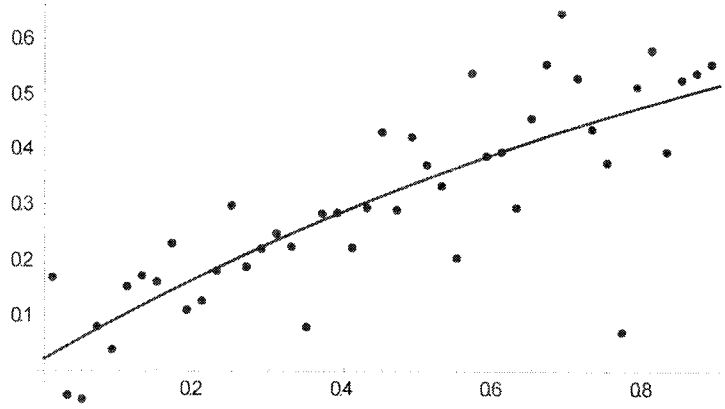
5. Estimating α(1) (asymptotic value of α) using MATHEMATICA. The data in vectors A1A, A1B, A2A and A2B are cut and pasted into the MATHEMATICA program (highlighted in yellow below), which produces estimates of the three regression parameters, and also plots the data and the fitted regression.

```
**********************************
(*  all samples , non_syn, freq diff = 1  *)
x={{0.01,0.1682589312},{0.03,-0.04384515007},{0.05,-
0.05024709661},{0.07,0.08014811619},{0.09,0.03959845735},{0.11,0.1521155831},{0.13,0.1
730674342},{0.15,0.1619181287},{0.17,0.2295005028},{0.19,0.1105263158},{0.21,0.1266399
695},{0.23,0.180632616},{0.25,0.29825443},{0.27,0.1890092879},{0.29,0.2209437387},{0.31
,0.247871517},{0.33,0.2245614035},{0.35,0.07985480944},{0.37,0.283110762},{0.39,0.28533
61728},{0.41,0.222109036},{0.43,0.2945553539},{0.45,0.4317251462},{0.47,0.2900165211},
{0.49,0.4218421053},{0.51,0.3719776715},{0.53,0.3337152845},{0.55,0.2039964318},{0.57,0
.5381578947},{0.59,0.3870853605},{0.61,0.3942847917},{0.63,0.2945553539},{0.65,0.45542
4275},{0.67,0.5552631579},{0.69,0.6475670308},{0.71,0.5305555556},{0.73,0.4359435173},
{0.75,0.3749644381},{0.77,0.07099415205},{0.79,0.5129072682},{0.81,0.5797762122},{0.83,
0.3960363873},{0.85,0.5256140351},{0.87,0.5385437277},{0.89,0.5552631579}};
x1=x[[All,{1,2}]];
nlm=NonlinearModelFit[x1,(1-(a+b Exp[-c y])),{a,b,c},y]
nlm[[1,2]]
Show[ListPlot[x1],Plot[nlm[x],{x,0,.9}]]
nlm[1]  (* value of alpha(1) *)

(* output *)
```

FittedModel[$0.958508 - 0.936118 e^{-0.835267 y}$]

{a->0.0414925,b->0.936118,c->0.835267}



0.552458

****

6. Estimating 95% Confidence Intervals for α(1). This is done by bootstrapping over SNPs within a SNP type (e.g. non-synonymous, synonymous, non-coding) using scripts NONLIN and BOOT1 (the former calls the latter). The script produces four matrices (MMAT1A, MMAT1B, MMAT2A, and MMAT2B) corresponding, respectively, to analyses of 1. Fixed differences, non-syn vs. syn; 2. Nearly fixed differences, non-syn vs. syn; 3. Fixed differences, non-coding vs. syn; and 4. Nearly fixed differences, non-coding vs. syn. These matrices are converted to character vectors for reading into MATHEMATIC. These vectors are saved as text files (APL Native Files), which are read by MATHEMATICA.

Each matrix consists of *n* columns and 1001 rows, where *n* is the number of allele frequency bins. The first column consists of the midpoints of each bin. Each of the remaining columns corresponds to one bootstrap sample. For each sample, the values are the α(d) values corresponding to the appropriate bin. The MATHEMATICA program calculates α(1) for each bootstrap sample and then calculates the confidence interval by ordering the α(1)'s and taking the 25th and 975th values.

```
**********************************
NONLIN
⍝ THIS PROGRAM RUNS NON-LINEAR REGRESSION ON BOOTSTRAP ALPHA DATA

⍝  NOTE: HAVE MIDPOINTS FROM PREVIOUSLY RUN SEPMERGEDNUCS

COUNT←+/MIDPOINTS<.9
MAT1A←MAT1B←MAT2A←MAT2B←(COUNT 1⍴MIDPOINTS)

MAXI←1000
I←0
RETI:I←I+1
```

```
TEST←(I÷100)=(⌈(I÷100))
⍞(TEST=1)/' ''I = '',I'

BOOT1     ⍝ CALL BOOT1 TO PROCESS EACH BOOTSTRAP SAMPLE

MAT1A←MAT1A,MATRIX[;2]
MAT1B←MAT1B,MATRIX[;3]
MAT2A←MAT2A,MATRIX[;4]
MAT2B←MAT2B,MATRIX[;5]

→(I<MAXI)/RETI

⍝ CONVERT MATRICES TO TEXT AND SAVE TO NATIVE FILES
MMAT1A←,⍕MAT1A
IND←(MMAT1A='¯')/⍳⍴MMAT1A
MMAT1A[IND]←'-'

'C:\MMAT1A' ⎕NCREATE ¯1
MMAT1A ⎕NAPPEND ¯1
⎕NUNTIE ¯1

MMAT1B←,⍕MAT1B
IND←(MMAT1B='¯')/⍳⍴MMAT1B
MMAT1B[IND]←'-'

'C:\MMAT1B' ⎕NCREATE ¯1
MMAT1B ⎕NAPPEND ¯1
⎕NUNTIE ¯1

MMAT2A←,⍕MAT2A
IND←(MMAT2A='¯')/⍳⍴MMAT2A
MMAT2A[IND]←'-'

'C:\MMAT2A' ⎕NCREATE ¯1
MMAT2A ⎕NAPPEND ¯1
⎕NUNTIE ¯1

MMAT2B←,⍕MAT2B
IND←(MMAT2B='¯')/⍳⍴MMAT2B
MMAT2B[IND]←'-'

'C:\MMAT2B' ⎕NCREATE ¯1
MMAT2B ⎕NAPPEND ¯1
⎕NUNTIE ¯1

⎕TCLF
'PROGRAM NONLIN FINISHED.  DATA STORED IN NATIVE FILES MMAT1A, MMAT1
B, MMAT2A, MMAT2B.'

**********************************
BOOT1

⍝  THIS PROGRAM BOOTSTRAPS THE ALPHA VALUES

DIM←⍴CNONFREQS
```

```
IND←?DIMρDIM
BCNONFREQS←CNONFREQS[IND]  ⍝ NOTE: CNONFREQS, ETC. FROM PREVIOUS RUN
 OF SEPMERGEDNUCS
BLNONFREQS←LNONFREQS[IND]

DIM←ρCSYNFREQS
IND←?DIMρDIM
BCSYNFREQS←CSYNFREQS[IND]
BLSYNFREQS←LSYNFREQS[IND]

DIM←ρCRFREQS
IND←?DIMρDIM
BCRFREQS←CRFREQS[IND]
BLRFREQS←LRFREQS[IND]

⍝ BIN THE FREQUENCIES   NOTE: BINS AND NUMBINS FROM PROGRAM SEPMERGE
DNUCS

NBINS←+/BCNONFREQS∘.>BINS
CNONTOT←+/[1]NBINS∘.=(⍳NUMBINS)

NBINS←+/BLNONFREQS∘.>BINS
LNONTOT←+/[1]NBINS∘.=(⍳NUMBINS)

NBINS←+/BCSYNFREQS∘.>BINS
CSYNTOT←+/[1]NBINS∘.=(⍳NUMBINS)

NBINS←+/BLSYNFREQS∘.>BINS
LSYNTOT←+/[1]NBINS∘.=(⍳NUMBINS)

NBINS←+/BCRFREQS∘.>BINS
CRTOT←+/[1]NBINS∘.=(⍳NUMBINS)

NBINS←+/BLRFREQS∘.>BINS
LRTOT←+/[1]NBINS∘.=(⍳NUMBINS)

NONTOT←CNONTOT+LNONTOT       ⍝ SUM NON-SYN SNPS OVER SPECIES
SYNTOT←CSYNTOT+LSYNTOT       ⍝ SUM SYN SNPS OVER SPECIES
RTOT←CRTOT+LRTOT             ⍝ SUM NON-CODING SNPS OVER SPECIES

⍝  CALCULATE ALPHA VECTORS FOR MESSER AND PETROV MK
⍝  NON-SYN FREQ DIFF = 1
DS←169
DN←190
ALPHA1A←1 - (DS÷DN)×NONTOT÷SYNTOT

⍝ NON-SYN FREQ DIFF [0.9, 1)
DS←699
DN←644
ALPHA1B←1 - (DS÷DN)×(NONTOT÷SYNTOT)

⍝ →DOWN3    ⍝  ONLY FOR CLOSE ALLOPATRIC

⍝ NON-CODING FREQ DIFF=1
DR←107
```

```
DS←169
ALPHA2A←1 - (DS÷DR)×(RTOT÷SYNTOT)

⍝ NON-CODING FREQ DIFF [0.9, 1)
DS←699
DR←333
ALPHA2B←1 - (DS÷DR)×(RTOT÷SYNTOT)

DOWN3:

⍝ CONVERT DATA TO MATRIX FOR SAS

MATRIX←(COUNT 1⍴MIDPOINTS),(COUNT 1⍴ALPHA1A),(COUNT 1⍴ALPHA1B),(COUN
T 1⍴ALPHA2A),(COUNT 1⍴ALPHA2B)
⍝ NOTE: COUNT IS FROM PROGRAM SEPMERGEDNUCS

⍝ CONVERT DATA TO MATRICES FOR MATHEMATICA

TMP←(COUNT 1⍴MIDPOINTS),(COUNT 1⍴ALPHA1A)
CONVMATH TMP
A1A←CONVDATA

TMP←(COUNT 1⍴MIDPOINTS),(COUNT 1⍴ALPHA1B)
CONVMATH TMP
A1B←CONVDATA

TMP←(COUNT 1⍴MIDPOINTS),(COUNT 1⍴ALPHA2A)
CONVMATH TMP
A2A←CONVDATA

TMP←(COUNT 1⍴MIDPOINTS),(COUNT 1⍴ALPHA2B)
CONVMATH TMP
A2B←CONVDATA
**********************************
```

The MATHEMATICA program for the estimation of bootstrapped CI's:

```
**********************************
SetDirectory["C:\Users\mrausher\Desktop"]
bootn=1000
x=ReadList["MMAT1A",Table[Number,{bootn+1}]];
list={};
For[i=2,i<(bootn+2),i++,x1=x[[All,{1,i}]];
 nlm=NonlinearModelFit[x1,(1-(a+b Exp[-c y])),{a,b,c},y];
 alpha1=nlm[1];AppendTo[list,alpha1](*Print[Show[ListPlot[x1],Plot[nlm[x],{x,0,.9}]]]*)];
list2=Sort[list];
confint={list2[[bootn .05]],list2[[bootn .95]]}
**********************************
```

VI. Analysis of admixture linkage disequilibrium (ALD). This is done with two scripts, ALD and COUNTALDB. For each *I. lacunosa* genome contig, ALD identifies synonymous SNPs on that contig and calculates pairwise distances and pairwise ald using the formula in the text. COUNTALDB bins these pairwise values according to pairwise distances and averages ald for all pairs within a bin. COUNTALDB calls script MAKEBINS, which defines the distance bins, and produces matrix 'ALDBINS', which is an N x 5 matrix with the following columns:

Col1 distance bin midpoint
Col2 ald(d)
Col3 count of SNP pairs in distance bin
Col4 mean absolute weightings of SNP pairs in distance bin
Col5 mean covariance between SNP pairs in distance bin

The versions of the scripts below are for calculating ald in the I. cordatotriloba sympatric samples.

```
**********************************
ALD
⍝ THIS PROGRAM CALCULATES ADMIXTURE LD FOR CORDAT SYMP POPULATION

SNPNUM←1↑⍴⍙SYNSCORES
ALD1←0 2⍴0
DIMSYMP←⍴⍙CSYMPINDEX
DATA←0 4⍴0
MAXI←840

I←0
RETI:I←I+1          ⍝ LOOP FOR CONTIGS
⎕TCLF
'I = ',I

TIG←⍙UNIQUETIGS[I]  ⍝ PICK OUT CONTIG NUMBER

SCINDEX←(⍙SYNSNPS[;1]=TIG)/⍳SNPNUM    ⍝  INDEX OF SNPS ON CONTIG

→(1≥⍴SCINDEX)/DOWN

SC←⍙SYNSCORES[SCINDEX;;]     ⍝ PICK SCORES FOR EACH SNP ON CONTIG
SPOS←⍙SYNSNPS[SCINDEX;3]     ⍝ PICK POSITIONS OF EACH SNP ON CONTIG

MAXSNP←⍴SCINDEX  ⍝ NUMBER OF SNPS ON CONTIG

J←0
RETJ:J←J+1    ⍝  SNP1 LOOP

SC1←SC[J;;]
SC1CA←SC1[⍙CALLO3;]
SC1LA←SC1[⍙LALLO3;]
UNSC1CS←SC1CS←SC1[⍙CSYMPINDEX;]

TEST←+/('ACGT'∈UNSC1CS)
```

```
COL1←SC1CA[;1]      ⍝ CONVERT GENOTYPES TO X'S FOR CORD ALLOPATRIC
IND1←COL1∈'ACGT'
COL2←SC1CA[;2]
IND2←COL2∈'ACGT'
IND3←IND1∧IND2
IND4←(IND3)/⍳⍴IND3
SC1CA←SC1CA[IND4;]

ALLELE1←SC1CA[1;1]

Q←SC1CA=ALLELE1
XCA←+/Q             ⍝  EACH GENOTYPE CONVERTED TO SUM OF NUMBER OF ALL
ELES GIVEN BY 'ALLELE'
FREQ1CA←(+/XCA)÷(2×⍴XCA)    ⍝  ALLELE FREQUENCY IN CORDAT ALLOPATRIC

COL1←SC1LA[;1]     ⍝  CONVERT GENOTYPES TO X'S FOR LAC ALLOPATRIC
IND1←COL1∈'ACGT'
COL2←SC1LA[;2]
IND2←COL2∈'ACGT'
IND3←IND1∧IND2
IND4←(IND3)/⍳⍴IND3
SC1LA←SC1LA[IND4;]
Q←SC1LA=ALLELE1
XLA←+/Q             ⍝  EACH GENOTYPE CONVERTED TO SUM OF NUMBER OF ALL
ELES GIVEN BY 'ALLELE'
FREQ1LA←(+/XLA)÷(2×⍴XLA)    ⍝  ALLELE FREQUENCY IN LAC ALLOPATRIC


COL1←SC1CS[;1]     ⍝  CONVERT TENOTYPES TO X'S FOR CORD SYMPATRIC
IND1←COL1∈'ACGT'
COL2←SC1CS[;2]
IND2←COL2∈'ACGT'
IND3←IND1∧IND2
IND4←(IND3)/⍳⍴IND3
SC1CS←SC1CS[IND4;]
Q←SC1CS=ALLELE1
XCS←+/Q             ⍝  EACH GENOTYPE CONVERTED TO SUM OF NUMBER OF ALL
ELES GIVEN BY 'ALLELE'
FREQ1CS←(+/XCS)÷(2×⍴XCS)    ⍝  ALLELE FREQUENCY IN CORD SYMPAtRIC

ALL1←,(SC1CA,[1]SC1LA),[1]SC1CS
TEST1←+/'ACGT'∈ALL1
→(TEST1≤1)/DOWN        ⍝ SKIP SNP IF NOT POLYMORPHIC

MEANX1←2×FREQ1CS

K←J
RETK:K←K+1    ⍝    SNP 2 LOOP

SC2←SC[K;;]
SC2CA←SC2[⍋CALLO3;]
SC2LA←SC2[⍋LALLO3;]
UNSC2CS←SC2CS←SC2[⍋CSYMPINDEX;]
```

```
COL1←SC2CA[;1]      ⍝ CONVERT GENOTYPES TO X'S FOR CORD ALLOPATRIC
IND1←COL1∈'ACGT'
COL2←SC2CA[;2]
IND2←COL2∈'ACGT'
IND3←IND1∧IND2
IND4←(IND3)/⍳⍴IND3
SC2CA←SC2CA[IND4;]

ALLELE2←SC2CA[1;1]

Q←SC2CA=ALLELE2
XCA2←+/Q            ⍝  EACH GENOTYPE CONVERTED TO SUM OF NUMBER OF AL
LELES GIVEN BY 'ALLELE'
FREQ2CA←(+/XCA2)÷(2×⍴XCA2)    ⍝  ALLELE FREQUENCY IN CORDAT ALLOPATRI
C

COL1←SC2LA[;1]     ⍝  CONVERT TENOTYPES TO X'S FOR LAC ALLOPATRIC
IND1←COL1∈'ACGT'
COL2←SC2LA[;2]
IND2←COL2∈'ACGT'
IND3←IND1∧IND2
IND4←(IND3)/⍳⍴IND3
SC2LA←SC2LA[IND4;]
Q←SC2LA=ALLELE2
XLA2←+/Q            ⍝  EACH GENOTYPE CONVERTED TO SUM OF NUMBER OF AL
LELES GIVEN BY 'ALLELE'
FREQ2LA←(+/XLA2)÷(2×⍴XLA2)    ⍝  ALLELE FREQUENCY IN LAC ALLOPATRIC

COL1←SC2CS[;1]     ⍝  CONVERT TENOTYPES TO X'S FOR CORD SYMPATRIC
IND1←COL1∈'ACGT'
COL2←SC2CS[;2]
IND2←COL2∈'ACGT'
IND3←IND1∧IND2
IND4←(IND3)/⍳⍴IND3
SC2CS←SC2CS[IND4;]
Q←SC2CS=ALLELE2
XCS2←+/Q            ⍝  EACH GENOTYPE CONVERTED TO SUM OF NUMBER OF AL
LELES GIVEN BY 'ALLELE'
FREQ2CS←(+/XCS2)÷(2×⍴XCS2)    ⍝  ALLELE FREQUENCY IN CORD SYMPARIC

ALL1←,(SC2CA,[1]SC2LA),[1]SC2CS
TEST2←+/'ACGT'∈ALL1
→(TEST2≤1)/DOWN3        ⍝ SKIP SNP IF NOT POLYMORPHIC
TEST←+/('ACGT'∈UNSC2CS)

MEANX2←2×FREQ2CS

⍝ CALCULATE COVAR(SNP1,SNP2)

SUM←0
COUNT←0
L←0
```

```
RETL:L←L+1
G1←UNSC1CS[L;]
G2←UNSC2CS[L;]

TEST←('⋆'∈(G1,G2))∨('.'∈(G1,G2))
→(TEST=1)/DOWN2

X1←+/G1=ALLELE1
X2←+/G2=ALLELE2
PROD←(X1-MEANX1)×(X2-MEANX2)
SUM←SUM+PROD
COUNT←COUNT+1

DOWN2:→(L<DIMSYMP)/RETL
COV←SUM÷(COUNT-1)

⍎(COV≥1.2)/'⎕TCLF ◇ ''COV GREATER THAN 1.2'' ◇ →0'

⍝  CALC ALPHA (ald)

W←(FREQ1CA-FREQ1LA)×(FREQ2CA-FREQ2LA)
ALPHA←COV×W
⍝  CALC DISTANCE

DIST←SPOS[K]-SPOS[J]

DATA←DATA,[1](ALPHA,DIST,COV,W)
DOWN3:→(K<MAXSNP)/RETK

DOWN:→(J<(MAXSNP-1))/RETJ

  →(I<MAXI)/RETI

'PROGRAM ALD FINISHED.  DATA IN VARIABLE ''DATA''.'
**********************************
COUNTALDB
⍝  THIS PROGRAM TAKES DATA CREATED BY PROGRAM ALD, BINS THEM, AND CA
LCULATES ALD(D) USING ALTERNATE FORMULA
⍝  BINS INCREASE IN SIZE EXPONENTIALLY
⍝  CORDAT SYMPATRIC SAMPLES

MAKEBINS 1.5

BINS←⍋BINS

MAXI←1↑⍴BINS
ALTMEANW←0⍴0

ALDBINS←0 5⍴0
X←⍋CALDDATAB2    ⍝ CHANGE FILE ACCORDINGLY
COUNTS←0⍴0

I←0
RETI:I←I+1
```

```
TEST←(I÷100)=(⌈(I÷100))
⍎(TEST=1)/'I'

COUNT←IND←+/(X[;2]≤(BINS[I;2]))
COUNTS←COUNTS,IND
PART←X[⍳IND;]

X←(IND,0)↓X
BINSIZE←BINS[I;2]-BINS[I;1]
A←(+/PART[;1])÷(COUNT)

MEANW←(+/|PART[;4]) ÷COUNT

REDPART←(|,PART[;4])

IND←(REDPART≠0)/⍳⍴REDPART
ALTM←(+/REDPART[IND])÷⍴IND
ALTMEANW←ALTMEANW,ALTM

MEANCOV←(+/,PART[;3])÷COUNT

MID←⌈(+/BINS[I;])÷2

ALDBINS←ALDBINS,[1](MID,A,COUNT,MEANW,MEANCOV)

DOWN:→(I<MAXI)/RETI

'PROGRAM COUNTALDA FINISHED.  DATA IN VARIABLE ''ALDBINS''.'
**********************************
MAKEBINS X

ΔBINS←1 2⍴(1,1000)
I←0
RETI:I←I+1
START←ΔBINS[I;2]+1
END←⌈(START-1)+1000×X*I
ΔBINS←ΔBINS,[1](START,END)
→(END<4500000)/RETI
**********************************
```

VII. Admixture and selfing rate estimates were performed using the script LIKS. It produces two matrices, ΔLNCLIK and ΔLNLLIK, that contain log-likelihood values for combinations of selfing rate and admixture proportion. Script MAKESAS2 converts these matrices into SAS form for graphics.

```
**********************************
LIKS X
⍝  THIS PROGRAM CALCULATES LIKELIHOOD OF ADMIXTURE PARAMETER HANIS E
T AL. 1986.  AM. J. PHYS. ANTHROPOOLOGY
⍝     70: 433-441
⍝ X IS ΔSCORES, OR ΔSYNSCORES, OR ΔNONSCORES, ETC.
⍝   THIS PROGRAM IS MODIFIED TO TAKE INTO ACCOUNT SELFING RATE, S

MS← .01 .1 .2 .3 .4 .5 .6 .7 .8 .9 .99   ⍝  VALUES M (GENE FLOW) CAN
 TAKE ON
⍝ MS←.21 .22 .23 .24 .25 .26 .27 .28 .29
⍝ MS←.31 .32 .33 .34 .35 .36 .37 .38 .39
⍝ MS←.71 .72 .73 .74 .75 .76 .77 .78 .79 .81 .82 .83 .84 .85 .86 .87
 .88 .89
SS←.1 .2 .3 .4 .5 .6 .7 .8 .9 .99
⍝ SS←.91 .92 .93 .94 .95 .96 .97 .98
⍝ SS←.81 .82 .83 .84 .85 .86 .87 .88 .89 .91 .92 .93 .94 .95 .96 .97
 .98

MAXII←⍴SS
MAXJ←1↑⍴MS
ΔLNCLIK←ΔLNLLIK←(MAXII,MAXJ)⍴0

II←0
RETII:II←II+1    ⍝  SELFING RATE LOOP
S←SS[II]

J←0
RETJ:J←J+1    ⍝ M LOOP (GENE FLOW)

M←MS[J]
⎕TCLF
'STARTING MVALUE = ',M,'    S VALUE = ',S

MAXI←1↑⍴X

CLIK←LLIK←1    ⍝ INITIAL LIKELIHOOD
LNCLIK←LNLLIK←0    ⍝  INITIAL LOG-LIKELIHOOD

I←0
RETI:I←I+1  ⍝  SNP LOOP

GENS←X[I;;]    ⍝ PICK GENOTYPES FOR SNP I

CALLO←GENS[ΔCALLO3;]  ⍝ PICK GENOTYPES FOR ALLOPATRIC I. CORDAT
LALLO←GENS[ΔLALLO3;]  ⍝ PICK GENOTYPES FOR ALLOPATRIC I. LAC
CSYMP←GENS[ΔCSYMPINDEX;]  ⍝ PICK GENOTYPES FOR SYMPATRIC I. CORDAT
LSYMP←GENS[ΔLSYMPINDEX;]  ⍝  PICK GENOTYPES FOR SYMPATRIC I. LAC
```

```
→(('★'∈,CALLO)∨('★'∈,LALLO)∨('★'∈,CSYMP)∨('★'∈,LSYMP))/DOWN1

TEST←(,CALLO),(,LALLO)
TEST2←+/'ACGT'∈TEST
→(TEST2=1)/DOWN1    ⍝  SKIP SNP IF CALLO AND LALLO FIXED FOR SAME ALL
ELE

IND←(,CALLO[;1]∈'ACGT')/⍳1↑⍴CALLO
CALLO←CALLO[IND;]     ⍝ GET RID OF '.'S AND '★'S
IND←(,LALLO[;1]∈'ACGT')/⍳1↑⍴LALLO
LALLO←LALLO[IND;]
IND←(,CSYMP[;1]∈'ACGT')/⍳1↑⍴CSYMP
CSYMP←CSYMP[IND;]
IND←(,LSYMP[;1]∈'ACGT')/⍳1↑⍴LSYMP
LSYMP←LSYMP[IND;]

ALLELE1←CALLO[1;1]

PCA←+/(ALLELE1=,CALLO) ÷(⍴,CALLO) ⍝ ALLELE 1 FREQ IN I. CORD ALLOPAT
RIC
QCA←1-PCA  ⍝ ALLELE 2 FREQ IN I. CORD ALLOPATRIC
PLA←+/(ALLELE1=,LALLO)÷(⍴,LALLO)
QLA←1-PLA
PCS←+/(ALLELE1=,CSYMP)÷(⍴,CSYMP)
QCS←1-PCS
PLS←+/(ALLELE1=,LSYMP)÷(⍴,LSYMP)
QLS←1-PLS


TEST←|(PCA-PLA)
→(TEST≥0.9)/DOWN1       ⍝ SKIP SNP IF ABSOLUTE FREQ DIFFERENCE IN ALL
OPATRY ≥ 0.9 ; CAN ALTER

MAXK←1↑⍴CSYMP

K←0
RETK:K←K+1    ⍝ LOOP FOR I. CORDAT SYMP INDIVIDUALS

IND←CSYMP[K;]  ⍝ PICK KTH INDIVIDUAL FROM I. CORDAT SYMP POPULATION
TEST←+/IND=ALLELE1

P1←(M×PCA) + ((1-M)×PLA)
Q1←1-P1
⍎(Q1<.0000001)/'Q1←0'
CORRECTION←S×P1×Q1÷(2×(1-(S÷2)))

⍎(TEST=2)/'P←(P1★2)+CORRECTION'
⍎(TEST=1)/'P←(2×P1×Q1)-(2×CORRECTION)'
⍎(TEST=0)/'P←(Q1★2)+CORRECTION '

CLIK←CLIK×P
LNCLIK←LNCLIK+⍟P

→(K<MAXK)/RETK
```

```
MAXL←1↑⍴LSYMP

L←0
RETL:L←L+1    ⍝ LOOP FOR I. LAC SYMP INDIVIDUALS

IND←LSYMP[L;]  ⍝ PICK KTH INDIVIDUAL FROM I. CORDAT SYMP POPULATION
TEST←+/IND=ALLELE1

P1←(M×PCA) + ((1-M)×PLA)
Q1←1-P1
CORRECTION←S×P1×Q1÷(2×(1-(S÷2)))

⍎(TEST=2)/'P←(P1*2)+CORRECTION'
⍎(TEST=1)/'P←(2×P1×Q1)-(2×CORRECTION)'
⍎(TEST=0)/'P←(Q1*2)+CORRECTION '

LLIK←LLIK×P
LNLLIK←LNLLIK+⍟P

→(L<MAXL)/RETL

DOWN1:→(I<MAXI)/RETI

ΔLNCLIK[II;J]←LNCLIK
ΔLNLLIK[II;J]←LNLLIK

⎕TCLF
'LNCLIK, LNLLIK ',LNCLIK,LNLLIK

→(J<MAXJ)/RETJ

→(II<MAXII)/RETII


⎕TCLF
'PROGRAM LIK FINISHED.  DATA IN VARIABLES  ΔLNCLIK, ΔLNLLIK'
**********************************


MAKESAS  X
⍝ THIS PROGRAM MAKES LIST OF M, S, AND LN LIKLIHOOD FOR SAS ANALYSIS
⍝  X IS ΔLNCLIK OR ΔLNLLIK FROM PROGRAM LIKS

⍝ MS← .01 .1 .2 .3 .4 .5 .6 .7 .8 .9 .99   ⍝  VALUES M (GENE FLOW) C
AN TAKE ON
⍝ MS←.21 .22 .23 .24 .25 .26 .27 .28 .29
⍝ MS←.31 .32 .33 .34 .35 .36 .37 .38 .39

MS←.71 .72 .73 .74 .75 .76 .77 .78 .79 .81 .82 .83 .84 .85 .86 .87 .
88 .89
⍝ SS←.1 .2 .3 .4 .5 .6 .7 .8 .9 .99
⍝ SS←.91 .92 .93 .94 .95 .96 .97 .98
SS←.81 .82 .83 .84 .85 .86 .87 .88 .89 .91 .92 .93 .94 .95 .96 .97 .
98
```

```
MAXI←⍴SS
MAXJ←⍴MS
DATA←0 3⍴0

I←0
RETI:I←I+1

J←0
RETJ:J←J+1

LINE←MS[J],SS[I],X[I;J]
DATA←DATA,[1]LINE

→(J<MAXJ)/RETJ
→(I<MAXI)/RETI

⎕TCLF
'PROGRAM MAKESAS FINISHED.  OUTPUT IN VARIABLE ''DATA''.'
```

Log-likelihoods of selfing rate estimates for allopatric samples wer e calculated using script SELFEST, followed by MAKESAS2 to convert t o SAS format for graphing.

```
SELFEST X

⍝ THIS PROGRAM CALCULATES LOG-LIKELIHOODS OF ESTIMATES OF SELFING RA
TES IN ALLOPATRIC
⍝  I. CORDAT AND I. LAC SAMPLES
⍝ X IS ΔSCORES, OR ΔSYNSCORES, OR ΔNONSCORES, ETC.

⍝SS←.01 .1 .2 .3 .4 .5 .6 .7 .8 .9 .99
SS←.1 .2 .3 .4 .5 .6 .7 .71 .72 .73 .74 .75 .76 .77 .78 .79 .80 .81
.82 .83 .84 .85 .86 .87 .88 .89 .9 .91 .92 .93 .94 .95 .96 .97 .98 .
99
⍝ SS←.91 .92 .93 .94 .95 .96 .97 .98
MAXII←1↑⍴SS
ΔSELFC←SELFL←0⍴0
ΔLNCLIK←ΔLNLLIK←0⍴0    ⍝  INITIAL LOG-LIKELIHOOD

II←0
RETII:II←II+1    ⍝  SELFING RATE LOOP
S←SS[II]

LNCLIK←LNLLIK←0

⎕TCLF
'STARTING S VALUE = ',S

MAXI←1↑⍴X

I←0
RETI:I←I+1  ⍝  SNP LOOP

GENS←X[I;;]   ⍝ PICK GENOTYPES FOR SNP I
```

```
CALLO←GENS[⍋CALLOCLOSE;]  ⍝ PICK GENOTYPES FOR ALLOPATRIC I. CORDAT
LALLOSAVE←LALLO←GENS[⍋LALLOCLOSE;]  ⍝ PICK GENOTYPES FOR ALLOPATRIC
I. LAC

IND←(,CALLO[;1]∈'ACGT')/⍳1↑⍴CALLO
CALLO←CALLO[IND;]     ⍝ GET RID OF '.'S AND '*'S
IND←(,LALLO[;1]∈'ACGT')/⍳1↑⍴LALLO
LALLO←LALLO[IND;]

TEST1←1↑⍴CALLO

⍎(TEST1=0)/'SKIPC←''Y'''
⍎(TEST1>0)/'SKIPC←''N'' ⋄ ALLELE1C←CALLO[1;1] ⋄ PCA←+/(ALLELE1C=,CAL
LO) ÷(⍴,CALLO)  ⋄ QCA←1-PCA  '
⍝ ALLELE 1 FREQ IN I. CORD ALLOPATRIC
⍝ ALLELE 2 FREQ IN I. CORD ALLOPATRIC

TEST2←1↑⍴LALLO
⍎(TEST2=0)/'SKIPL←''Y'''
⍎(TEST2>0)/'SKIPL←''N'' ⋄ ALLELE1L←LALLO[1;1]  ⋄ PLA←+/(ALLELE1L=,LA
LLO)÷(⍴,LALLO) ⋄ QLA←1-PLA '

MAXK←1↑⍴CALLO

→(SKIPC='Y')/DOWN2  ⍝ SKIP SNP IF NO INDIVIDUALS

TEST←+/'ACGT'∈,CALLO
→(TEST=1)/DOWN2   ⍝ SKIP SNP IF NO VARIATION

K←0
RETK:K←K+1   ⍝ LOOP FOR I. CORDAT ALLO INDIVIDUALS

IND←CALLO[K;]  ⍝ PICK KTH INDIVIDUAL FROM I. CORDAT SYMP POPULATION

TEST←+/IND=ALLELE1C
CORRECTION←S×PCA×QCA÷(2×(1-(S÷2)))

⍎(TEST=2)/'P←(PCA*2)+CORRECTION'
⍎(TEST=1)/'P←(2×PCA×QCA)-(2×CORRECTION)'
⍎(TEST=0)/'P←(QCA*2)+CORRECTION'

LNCLIK←LNCLIK+⍟P

→(K<MAXK)/RETK

DOWN2:MAXL←1↑⍴LALLO

→(SKIPL='Y')/DOWN1

TEST←+/'ACGT'∈,LALLO
→(TEST=1)/DOWN1

L←0
RETL:L←L+1   ⍝ LOOP FOR I. LAC ALLO INDIVIDUALS
```

```
IND←LALLO[L;]  ⍝ PICK KTH INDIVIDUAL FROM I. CORDAT SYMP POPULATION

TEST←+/IND=ALLELE1L

CORRECTION←S×PLA×QLA÷(2×(1-(S÷2)))

⍎(TEST=2)/'P←(PLA*2)+CORRECTION'
⍎(TEST=1)/'P←(2×PLA×QLA)-(2×CORRECTION)'
⍎(TEST=0)/'P←(QLA*2)+CORRECTION '

LNLLIK←LNLLIK+⍟P

→(L<MAXL)/RETL

DOWN1:→(I<MAXI)/RETI
ΔLNCLIK←ΔLNCLIK,LNCLIK
ΔLNLLIK←ΔLNLLIK,LNLLIK

⎕TCLF
'LNCLIK, LNLLIK ',LNCLIK,LNLLIK

→(II<MAXII)/RETII

⎕TCLF
'PROGRAM SELFEST FINISHED.  DATA IN VARIABLES  ΔLNCLIK, ΔLNLLIK'
**************************************************

MAKESAS2
⍝  THIS TAKES ΔLNCLIK AND ΔLNLLIK FROM PROGRAM 'SELFEST' AND MAKES D
ATA FOR SAS
⍝  VARIABLE SS WAS JUST FORMED BY 'SELFEST'

MAXI←⍴SS
DATA←0 3⍴0
I←0
RETI:I←I+1
S←SS[I]
LINE←S,ΔLNCLIK[I],ΔLNLLIK[I]
DATA←DATA,[1]LINE
→(I<MAXI)/RETI

'PROGRAM MAKESAS2 FINISHED.  OUTPUT IN VARIABLE ''DATA''.'
```

VIII. Simulating gene flow. The script BREAKPOS divides all contigs into blocks of size 10 kb. This is run first. It produces the variable ΔBLOCKS, each row of which contains the first and last position of block's SNPs in ΔSCORES.

The script SWAPFIT3 swaps different proportions of alleles in 100 kb contig segments from allopatric *I. lacunosa* into allopatric *I. cordatotriloba* samples to create simulated *I. cordatotriloba* sympatric samples. For each proportion, it evaluates the sum of squared differences between the simulated allele frequencies and observed allele frequencies in the actual *I. cordatotriloba* sympatric samples. The program also calculates mean of sympatric between-species π across 20 replicate simulations for each proportion. The proportion with the value of π closest to the observed value is taken as the best estimate of the true proportion.

```
BREAKPOS
⍝ THIS DIVIDES ALL CONTIGS INTO BLOCKS OF SIZE BLOCKSIZE KB OR LESS
BLOCKSIZE← 100000  ⍝ SIZE OF BLOCK IN KILOBASES
ΔBLOCKS←0 2⍴0            ⍝ WILL HOLD START AND END INDEX OF BLOCKS
DIM←⍴ΔUNIQUETIGS        ⍝ NUMBER OF UNIQUE CONTIGS
I←0
RETI:I←I+1
CONTIGNUMS←(ΔUNIQUETIGS[I]=ΔTIGS)/⍳⍴ΔTIGS     ⍝  PICK LOCUS NUMBERS F
OR CONTIG I
POS←ΔPOS[CONTIGNUMS]                          ⍝  PICK POSITIONS ON CO
NTING

RET:STARTNUM←CONTIGNUMS[1]                    ⍝ THIS LOOOP DIVIDES CO
NTIG INTO 100 KB BLOCKS
STARTPOS←POS[1]                               ⍝ FIRST CONTIG POSITION
 OF BLOCK

IND3←POS<(STARTPOS+BLOCKSIZE)                    ⍝ PICK OUT ALL POSIT
IONS WITHIN BLOCKSIZE
NUM←+/IND3                                       ⍝  NUM IS LIST OF PO
SITIONS IN BLOCK

BLOCKNUMS←NUM↑CONTIGNUMS                         ⍝  TAKE LOCUS NUMBER
S FROM CONTIGNUMS
BLOCK←(1↑BLOCKNUMS), (¯1↑BLOCKNUMS)              ⍝  DEFINE FIRST AND
LAST LOCI NUMS FOR BLOCK
ΔBLOCKS←ΔBLOCKS,[1]BLOCK                           ⍝  ADD BLOCK TO BL
OCKS
CONTIGNUMS←NUM↓CONTIGNUMS                          ⍝ DROP LOCI NUMBER
S FROM CONTIGNUMS
POS←NUM↓POS                                        ⍝ DROP POSITIONS F
ROM ΔPOS
→(0<⍴CONTIGNUMS)/RET
→(I<DIM)/RETI
**********************************
```

Program SEPSCORES creates an index, ΔFIXEDSCORES that indicates whether a SNP is highly diverged or less diverged in allopatry for either known or close allopatric sites

```
SEPSCORES  X

⍝ THIS FUNCTION IDENTIFIES SNPS THAT ARE HIGHLY DIVERGENT (FREQ DIFF
 IN ALLOPATRY ≥ 0.9) OR LESS DIVERGENT (FREQ DIFF IN ALLOPATRY < 0.9
)
⍝  IT CREATES A VARABLE ⍙FIXEDSCORES THAT HAS 2 ROWS AND 66,720 COLU
MNS, ONE COLUMN FOR EACH SNP
⍝  A 1 IN FIRST ROW INDICATES THAT SNP IS HIGHLY DIVERGENT FOR KNOWN
 ALLOPATRIC SAMPLES
⍝  A 1 IN SECOND ROW INDICATES THAT SNP IS HIGHLY DIVERGENT FOR CLOS
E ALLOPATRIC SAMPLES
⍝  A 0 IN EITHER ROW INDICATES SNP IS LESS DIVERGENT
⍝    TYPICALLY X WILL BE ⍙SCORES, BUT IT COULD BE A SUBSET OF ⍙SCORE
S (E.G. SYNONYMOUS SNPS, NON-SYNONYMOUS SNPS, ETC.)

⍙FIXEDSCORES←2 0⍴0

MAXI←1↑⍴X
I←0          ⍝  SNP LOOP
RETI:I←I+1

SCORES←X[I;;]

CALLOK←,SCORES[⍙CALLO3;]          ⍝ PICK OUT GENOTYPES CORRESPONDING
TO I. CORDAT KNOWN ALLOPATRIC SITES

LALLOK←,SCORES[⍙LALLO3;]          ⍝ PICK OUT GENOTYPES CORRESPONDING
TO I. LAC KNOWN ALLOPATRIC SITES

CALLOC←,SCORES[⍙CALLOCLOSE;]      ⍝  DITTO FOR CLOSE ALLOPATRIC SITES
LALLOC←,SCORES[⍙LALLOCLOSE;]      ⍝  DITTO

CALLOK1←(CALLOK∈'ACGT')/CALLOK    ⍝ ELILMINATE MISSING DATA
LALLOK1←(LALLOK∈'ACGT')/LALLOK
CALLOC1←(CALLOC∈'ACGT')/CALLOC
LALLOC1←(LALLOC∈'ACGT')/LALLOC

ALLELE←1↑CALLOK1                  ⍝ DESIGNATE FOCAL ALLELE FOR FREQUE
NCY CALCULATIONS

DIFF1←|((+/CALLOK1=ALLELE)÷(⍴CALLOK1))-((+/LALLOK1=ALLELE)÷(⍴LALLOK1
)) ⍝ KNOWN ALLOPATRIC FREQUENCY DIFFERENCE

DIFF2←|((+/CALLOC1=ALLELE)÷(⍴CALLOC1))-((+/LALLOC1=ALLELE)÷(⍴LALLOC1
)) ⍝ CLOSE ALLOPATRIC FREQUENCY DIFFERENCE

ADD←2  1⍴0   ⍝ INITIALIZE VARIABLE TO HOLD 1 0R 0 VALUES

⍎(DIFF1≥0.9)/'ADD[1;1]←1'       ⍝  IF KNOWN ALLOPATRIC DIFFERENCE ≥0.
9, PLACE 1 IN FIRST ROW OF 'ADD'
```

```
⍝(DIFF2≥0.9)/'ADD[2;1]←1'      ⍝ IF CLOSE ALLOPATRIC DIFFERENCE ≥0.
9, PLACE 1 IN SECOND ROW OF 'ADD'

ΔFIXEDSCORES←ΔFIXEDSCORES,ADD  ⍝ APPEND RESULTS TO ΔFIXEDSCORES

→(I<MAXI)/RETI

⎕TCLF
'PROGRAM SEPSCORES FINISHED. DATA IN VARIABLE ''ΔFIXEDSCORES''.'

**********************************************************************
```

Program MIXPI. This program simulates gene flow by replacing a proportion of the genotypes in the "newly sympatric" *I. cordatotriloba* population with randomly chosen genotypes from the newly sympatric *I. lacunosa* population. Data saved in variable OUTPUT.

```
MIXPI

⍝ THIS FUNCTION SIMILATES GENE FLOW AND CALCULATES RESULTING PI FOR
DIFFERENT VALUES OF PROPORTIONS
⍝   ADMIXTURE (PROPORTION OF ALLELES REPLACED IN ''NEWLY SYMPATRIC''
 POPULATION).

PROPNUMS←.45 .46 .47 .48 .49 .51 .52 .53 .54 .55      ⍝ ADMIXTURE PR
OPORTIONS TO USE; CAN CHANGE ACCORDINGLY
OUTPUT←0 12⍴0

MAXII←⍴PROPNUMS
II←0                ⍝ PROPORTION LOOP
RETII:II←II+1
PROP←PROPNUMS[II]   ⍝ PICK ADMIXTURE PROPORTION

MAXJ←20
J←0
RETJ:J←J+1   ⍝ REPLICATE LOOP FOR A GIVEN ADMIXTURE PROPORTION
II,J,PROP

SWAPTIG3MA    ⍝ PRODUCES PSEUDOSCORES BY SWAPPING GENOTYPES FROM ALLO
PATRIC TO SYMPATRIC POPULATION

PICALC2B PSEUDOSCORES  ⍝  CALCULATES ΔPI

PICONTRAST2            ⍝  CALCULATES AVERAGE PI BETWEEN SYMPATRIC SA
MPLES  (CONTAINED IN ΔOBSPIVECTOR)

OUTPUT←OUTPUT,[1](PROP,ΔOBSPIVECTOR)    ⍝ CATENATES ΔOBSPIVECTOR TO O
UTPUT

→(J<MAXJ)/RETJ

→(II<MAXII)/RETII


⎕TCLF
```

```
'PROGRAM MIXPI FINISHED.  DATA IN VARIABLE ''OUTPUT''.'

**********************************************************
```

Program SWAPTIG3MA. This program is called by program MIXPI. It performs the genome swapping so simulate gene flow from *I. lacunosa* to *I. cordatotriloba* in sympatry. It produces a variable PSEUDOSCORES

```
SWAPTIG3MA

⍝ THIS FUNCTION SIMULATES GENE FLOW FROM I. LACUNOSA INTO I. CORDATO
TRILOBA IN SYMPATRY.
⍝    IT FIRST CREATES 'NEW' SYMPATRIC POPULATION BY SUBSTITUTING RAND
ROM DRAWS FORM CORD ALLO AND LAC ALLO
⍝    INTO CORD SYMP AND LAC SYMP. SUBSTITUTIONS ARE BY BLOCK
⍝    IT THE SUBSTITUES A RANDOMLY CHOSEN NUMBER OF LAC ALLO INDIVIDUA
LS INTO THE NEW CORD SYMPATRIC
⍝    POPULATION, REPLACING A RANDOMLY CHOSEN SET OF INDIVIDUALS.
⍝    THE NUMBER OF INDIVIDUALS SUBSTITUTED IS DETERMINED BY 'PROP'


PSEUDOSCORES←0 62 2⍴' '    ⍝  NEW VARIABLE TO HOLD DATA
DIM1←1↑⍴△SCORES
BLOCKDIM←1↑⍴△BLOCKS        ⍝  NUMBER OF BLOCKS

I←0                        ⍝ BLOCK LOOP
RETI:I←I+1


BLOCK←△BLOCKS[I;]  ⍝ CHOOSE ENDPOINTS OF BLOCK I
LOCNUMS←(BLOCK[1]-1)+⍳(1+BLOCK[2]-BLOCK[1])    ⍝ CALCULATE LOCI NUMBE
RS

PARTSCORES←△SCORES[LOCNUMS;;]

⍝ NEXT PART SUBSTITUTES L ALLO INTO L SYMP
LALLOSCORES←PARTSCORES[;△LALLO3;]  ⍝ NOTE CAN CHANGE TO △LALLOCLOSE
RAND←?13⍴16       ⍝ INDEX FOR RANDOMLY CHOSEN LAC ALLO INDIVIDUALS
PARTSCORES[;△LSYMPINDEX;]←LALLOSCORES[;RAND;]    ⍝ MAKE SUBSTITUTION

⍝ NEXT PART SUBSTITUTES C ALLO INTO C SYMP
CALLOSCORES←PARTSCORES[;△CALLO3;]
RAND2←?11⍴14
PARTSCORES[;△CSYMPINDEX;]←CALLOSCORES[;RAND2;]

⍝ NEXT PART SUBSTITUTES RANDOM INDIVIDUAL FROM LALLO INTO CSYMP
NUMSUB←⌊/(( ⌊((PROP×(⍴△CSYMPINDEX))+((?10000)÷10000))),(⍴△CSYMPINDEX
))  ⍝  THIS IS NUMBER OF INDIVIDUALS IN 'NEW' LSYMP TO CHOOSE
RAND3←?NUMSUB⍴13              ⍝ RANDOM INDEX FOR PICKING SUBS
SUBS←PARTSCORES[;△LSYMPINDEX[RAND3];]  ⍝ PICK SUBS FROM NEW LSYMP
RAND4←NUMSUB?11    ⍝  RANDOM INDEX FOR CSYMP INDIVIDS TO BE REPLACED
PARTSCORES[;△CSYMPINDEX[RAND4];]←SUBS
```

```
PSEUDOSCORES←PSEUDOSCORES,[1]PARTSCORES ⍝ ADD TO PSEUDOSCORES

→(I<BLOCKDIM)/RETI

IND←(~∆FIXEDSCORES[2;])/⍳1↑⍴PSEUDOSCORES      ⍝  PICKS OUT FIXED AND
NEARLY FIXED SCORES

PSEUDOSCORES←PSEUDOSCORES[IND;;]

**********************************************************
```

Program PICALC2B. This program is called by program MIXPI. It calculates pairwise between-species PI* for sympatric samples and saves them in variable ΔPI2

```
PICALC2B X;I

⍝ THIS PROGRAM CALCULATES PAIRWISE PI VALUES FOR EITHER THE FULL DAT
ASET (ALL SNPS)
⍝   OR FOR SUBSETS OF DATA (E.G. SYNONYMOUS, NON-SYNONYMOUS SITES)
⍝ IT PRODUCES A MATRIX ∆PI2 THAT HAS THE AVERAGE PAIRWISE PI VALUES
FOR EACH PAIR OF SAMPLES
⍝  PROGRAM READS IN VARIABLE X (PSEUDOSCORES) CREATED BY SWAPTIG3MA

PICUM←62 62⍴0   ⍝ SET UP PI MATRIX- WILL EVENTUALLY HAVE NUMBER OF P
AIRWISE DIFFERENCES ACROSS ALL VARIABLE SNPS

PICOUNT←62 62⍴0   ⍝ SET UP MATRIX FOR CUMULATIVE COUNTS (EXCLUDES MI
SSING VALUES)

DIM←1↑⍴X
I←0
RETI:I←I+1        ⍝ SNP LOOP

⍝  PICK DATA FOR NEXT SNP--STRIP OFF LEADING INFO AND LEAVE JUST NUC
LEOTIDES FOR DIFFERENT
⍝       SAMPLES.  TEMP5 HAS FORMAT G/G G/G G/C . . .

SCORES3←X[I;;]            ⍝  NOTE: MODIFY WHICH ∆SCORES TO USE (E.G.
ALL, SYNONYMOUS, ETC.)

⍝ CALCULATE PAIRWISE PI VALUES FUR CURRENT SNP
TEMP7←SCORES3∘.=SCORES3
TEMP8←+/[2]TEMP7
TEMP9←(+/[3]TEMP8)÷4
TEMP9←1-TEMP9
TIND←(SCORES3[;1]='.')/⍳62
COUNTMAT←62 62⍴1
COUNTMAT[TIND;]←0
COUNTMAT[;TIND]←0
PI←TEMP9×COUNTMAT

⍝ ADD CURRENT PAIRWISE PI VALUES TO CUMULATIVE VALUES
PICUM←PICUM+PI
PICOUNT←PICOUNT+COUNTMAT
```

```
→(I<DIM)/RETI

MAXPICOUNT←⌈/,,PICOUNT
CORRCOUNT←MAXPICOUNT÷PICOUNT
⍝ ΔPI2←PICUM×CORRCOUNT÷DIVIDER
⍝ΔPI2←PICUM÷DIVIDER
ΔPI2←PICUM÷PICOUNT

□TCLF
'PROGRAM PICALC2 FINISHED.'
'PAIRWISE PI VALUES IN VARIABLE ''ΔPI2''.'
```

****

Program PICONTRAST2. This program is called by MIXPI and calculates *average* between-species sympatric PI (PI*) values and saves them in ΔOBSPIVECTOR.

```
PICONTRAST2
⍝ THIS PROGRAM CALCULATES PI WITHIN AND BETWEEN SYMP AND ALLOPATRIC
SAMPLES
⍝    IT USES THE ΔPI2 MATRIX CALCULATED BY 'PICALC2'

MASK4←62 62⍴1

I←0
RETI:I←I+1
MASK4[I;I]←0
→(I<62)/RETI

⍝PI←MASK4×ΔPI×(1↑⍴ΔSCORES2)÷ΔTOTNUC  ⍝ THIS STEP CONVERTS THE PI VAL
UES CALCULATED BY 'PICALC2'
                                    ⍝      TO TRUE PI VALUES
PI←MASK4×ΔPI2

PIBETWALLO←PI[ΔCALLO3;ΔLALLO3]
PIBETWALLOCLOSE←PI[ΔCALLOCLOSE;ΔLALLOCLOSE]
PIBETWSYMP←PI[ΔCSYMPINDEX;ΔLSYMPINDEX]

PICBETWALLOSYMP←PI[ΔCALLO3;ΔCSYMPINDEX]
PILBETWALLOSYMP←PI[ΔLALLO3;ΔLSYMPINDEX]

PICBETWALLOCLOSESYMP←PI[ΔCALLOCLOSE;ΔCSYMPINDEX]
PILBETWALLOCLOSESYMP←PI[ΔLALLOCLOSE;ΔLSYMPINDEX]

AVEBETWSYMP←(+/+/PIBETWSYMP)÷((⍴ΔCSYMPINDEX)×((⍴ΔLSYMPINDEX)))
AVEBETWALLO←(+/+/PIBETWALLO)÷((⍴ΔCALLO3)×((⍴ΔLALLO3)))
AVEBETWALLOCLOSE←(+/+/PIBETWALLOCLOSE)÷((⍴ΔCALLOCLOSE)×((⍴ΔLALLOCLOS
E)))

AVECBETWALLOSYMP←(+/+/PICBETWALLOSYMP)÷((⍴ΔCALLO3)×(⍴ΔCSYMPINDEX))
AVELBETWALLOSYMP←(+/+/PILBETWALLOSYMP)÷((⍴ΔLALLO3)×(⍴ΔLSYMPINDEX))

AVECBETWALLOCLOSESYMP←(+/+/PICBETWALLOCLOSESYMP)÷((⍴ΔCALLOCLOSE)×(⍴Δ
CSYMPINDEX))
```

```
AVELBETWALLOCLOSESYMP←(+/+/PILBETWALLOCLOSESYMP)÷((ρ⍙LALLOCLOSE)×(ρ⍙
LSYMPINDEX))

⍙BETWDIFF1←AVEBETWALLO-AVEBETWSYMP
⍙BETWDIFF2←AVEBETWALLOCLOSE-AVEBETWSYMP
⍙BETWDIFF3←AVECBETWALLOSYMP-AVELBETWALLOSYMP
⍙BETWDIFF4←AVECBETWALLOCLOSESYMP-AVELBETWALLOCLOSESYMP

⍙OBSPIVECTOR←AVEBETWALLO,AVEBETWALLOCLOSE,AVEBETWSYMP,⍙BETWDIFF1,⍙BE
TWDIFF2,AVECBETWALLOSYMP,AVELBETWALLOSYMP,AVECBETWALLOCLOSESYMP,AVEL
BETWALLOCLOSESYMP,⍙BETWDIFF3,⍙BETWDIFF4 ⍝ OBSERVED VALUES

⎕TCLF
'BETWEEN SPECIES COMPARISIONS'
⎕TCLF
'a. AVERAGE PI BETWEEN KNOWN ALLO: ',AVEBETWALLO
'b. AVERAGE PI BETWEEN CLOSE ALLO: ',AVEBETWALLOCLOSE
'c. AVERAGE PI BETWEEN SYMP:       ',AVEBETWSYMP
'd. DIFFERENCE a. - c.:            ',(⍙BETWDIFF1)
'e. DIFFERENCE b. - c.:            ',(⍙BETWDIFF2)
⎕TCLF
'WITHIN SPECIES COMPARISONS'
⎕TCLF
'f. I. CORD BETW ALLO(KNOWN) AND SYMP ',(AVECBETWALLOSYMP)
'g. I. LAC  BETW ALLO(KNOWN) AND SYMP ',(AVELBETWALLOSYMP)
'h. I. CORD BETW ALLO(CLOSE) AND SYMP ',(AVECBETWALLOCLOSESYMP)
'i. I. LAC  BETW ALLO(CLOSE) AND SYMP ',(AVELBETWALLOCLOSESYMP)

'j. DIFFERENCE f. - g.:               ',(⍙BETWDIFF3)
'k. DIFFERENCE h. - i.:               ',(⍙BETWDIFF4)


⎕TCLF
'PROGRAM PICONTRAST2 FINISHED'

**********************************************************************
```

Program SUMMARIZEPI. This program takes variable OUTPUT produced by program MIXPI and calculates mean PI* and its standard deviation for each value of admixture proportion.

```
SUMMARIZEPI X
⍝ THIS FUNCTION READS IN X, WHICH IS VARIABLE "OUTPUT" FROM MIXPI.
⍝  IT PRODUCES VARIABLE DATA, WHICH HAS 3 COLUMNS:

⍝          ADMIXTURE PROP        MEAN PI*     SD PI*

⍝  EACH ROW CORESPONDS TO A DIFFERENT ADMIXTURE PROPORTION

DATA←0 3ρ0
ROWS←1↑ρX
CATS←ROWS÷20

I←0
RETI:I←I+1
IND←((I-1)×20)+⍳20
```

```
PART←X[IND;]
PROP←PART[1;1]
B←,PART[;4]
MEAN←+/B÷20
VAR←(+/(B-MEAN)*2)÷19
DATA←DATA,[1](PROP,MEAN,(2×VAR*.5))
→(I<CATS)/RETI

□TCLF
'PROGRAM SUMMARIZEPI FINISHED.  DATA IN VARIABLE ''DATA''.'

**************************************************
```

Program PISIMBOOT. This program bootstraps between-species sympatric PI values of observed data to estimate 95% credible intervals for PI for either highly-divergent or less-divergent SNPs. Bootstrap PI values accumulated in variable ΔPIBETW. Mean and credible interval calculated manually from this variable.

```
PISIMBOOT

⍝ THIS PROGRAM BOOTSTRAPS THE OBSERVED BETWEEN-SPECIES SYMPATRIC PI
SCORES FOR
    ⍝  EITHER HIGHLY DIVERGED OR LESS-DIVERGED SNPS

SCORES←ΔSCORES  ASSIGN SNP GENOTYPES TO VARIABLE 'SCORES'

ΔPIBETW←0⍴0      ⍝  INITIALIZE VARIABLE TO CONTAIN PI VALUES FOR DIFF
ERENT SNPS

IND←(~ΔFIXEDSCORES[2;])/⍳1↑⍴SCORES      ⍝  INDEX FOR FIXED AND NEARLY
 FIXED SCORES
    ⍝  OR LESS DIVERGENT SCORES (DEPENDS IF '~' PRESENT BEFORE 'ΔFIX
EDSCORES')
    ⍝ ROW 1 CORRESPONDS TO KNOWN ALLOPATRIC SAMPLES, ROW 2 TO CLOSE
ALLOPATRIC SAMPLES

SCORES←SCORES[IND;;]  ⍝ PICKS OUT HIGHLY DIVERGENT OR LESS DIVERGENT
 SCORES (GENOTYPES)

CSYMP←ΔCSYMPINDEX      ⍝ INDEX OF CORDAT SYMPATRIC SAMPLES
LSYMP←ΔLSYMPINDEX      ⍝ INDEX OF LAC SYMPATRIC SAMPLES

II←0
RETII:II←II+1             ⍝ SNP LOOP
TEST←(II÷100)=(⌊II÷100)
⍎(TEST=1)/'II'

TSCORES←SCORES            ⍝ BOOTSTRAP LOOP-- BOOTSTRAPPING OVER SAMPL
ES
IND1←CSYMP[?(⍴CSYMP)⍴⍴CSYMP]          ⍝ INDEX FOR  CORDAT SAMPLES PI
CKED RANDOMLY WITH REPLACEMENT
TSCORES[;CSYMP;]←SCORES[;IND1;]       ⍝ REPLACE OBS SAMPLE GENOTYPES


IND2←LSYMP[?(⍴LSYMP)⍴⍴LSYMP]          ⍝  DITTO FOR LAC SAMPLES
```

```
TSCORES[;LSYMP;]←SCORES[;IND2;]

PICALC2A TSCORES        ⍝ PRODUCES ΔPI2

PICONTRAST2B            ⍝  CALCULATES AVERAGE BETWEEN-SPECIES PI VALUE
S

PIBETW←ΔOBSPIVECTOR[3]

ΔPIBETW←ΔPIBETW,PIBETW      ⍝ APEND PI VALUE TO VECTOR OF PI VALUES.


→(II<1000)/RETII

⍝  MANUALLY COMPUTE MEAN AND 95 PERCENT CREDIBLE INTERVAL FOR PI
```