APL Pipeline for Transcriptome Project

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 XXX 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.

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

© THIS PROGRAM READS IN DATASET JG1 AND CALCULATES ALLELE COUNTS SEP
ARATELY FOR
©    THE DIFFERENT POPULATIONS
© ALSO, MAKES THREE VARIABLES:
‘SCORES„0 62 2½' '  © VARIABLE TO HOLD GENOTYPE CALLS
‘TIGS„0½0           © VARIABLE TO HOLD CORRESPONDING CONTIG NUMBERS
‘POS„0½0            © VARIABLE TO HOLD CORRESPONDING GENOME POSITION

'C:\JG1' ŒNTIE ¯1
SIZE„ŒNSIZE ¯1
FILE„ŒNREAD (¯1,82,(SIZE+1000),0)   ©  READ IN FILE
ŒNUNTIE ¯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„0½0    © INITIALIZE VARIABLES FOR INDIVIDUAL
SAMPLE HETEROZYGOSITIES
TOTC„TOTL„0 5½0                © SET UP VARIABLES FOR TOTAL COUNTS FO
R C AND L FOR 5 NUCLEOTIDES (INCLUDING *)

CINDEX„(1+¼31),62    ©  SET UP INDICIES FOR ROWS OF DATA2 TO DISTING
UISH SAMPLES FROM DIFFERENT POPULATIONS
AUSTINDEX„1
LINDEX„32+¼29


I„0
RETI:I„I+1

TEMP6„((I÷1000)-(—(I÷1000)))   © REPORT SNP NUMBER EVERY 100 SNPS
—(TEMP6=0)/'ŒTCLF ª 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 ŒSS 'tig'
TEMP3„TEMP2/¼½TEMP2
TEMP4„(TEMP3[2]-1)†TEMP
TEMP5„(TEMP3[1]-1)‡TEMP4

©  IDENTIFY NUCLEOTIDE IN REFERENCE GENOME (NOT CURRENTLY USED^
IND0„(TEMP5=ŒAV[10])/¼½TEMP5
TIG„IND0[1]†TEMP5
TIG2„(TIG¹'0123456789')/TIG
TIG3„ŒFI TIG2
‘TIGS„‘TIGS,TIG3

POS„¯1‡(IND0[1]‡IND0[2]†TEMP5)
POS2„ŒFI POS
‘POS„‘POS,POS2

©  REFORMAT TEMP5 AS 62 X 3 MATRIX OF NUCLEOTIDES IN FORMAT G/G
IND1„TEMP5='/'
IND2„IND1/¼½IND1
SCORES„(IND2[1]-2)‡TEMP5
SCORES2„62 4½SCORES
SCORES3„SCORES2[;¼3]
CHECK„(+/SCORES3[;2]='/')=62    © CHECK TO ENSURE PROGRAM IS NOT OUT
 OF ALIGNMENT
—(CHECK=0)/'''BAD DATA FOR SNP '',I ª ŒTCLF ª SCORES3 ª…0'
‘SCORES„‘SCORES,[1]SCORES3[;1 3]

DATA2„(TEMP3[2]-1)‡DATA2       ©  DROP CURRENT SNP FROM DATA2

TEST„+/(DATA2 ŒSS 'tig')

…(TEST>1)/RETI
ŒTCLF
'PROGRAM COMPLETE'
'    HETEROZYGOSITIES IN VARIABLES CHET, LHET, AUSTHET, AND WHET'
'    NUCLEOTIDE COUNTS IN VARIABLES TOTC AND TOTL'
'    CALLS IN VARIABLE '''‘SCORES'''
'    CORRESPONDING CONTIGS IN VARIABLE '''‘TIGS'''
'    CORRESPONDING CONTIG POSITION IN VARIABLE '''‘POS'''
'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„0½0' 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„½GTF
'C:\LACGFF1' ŒNTIE ¯1
GTF„GTF,ŒNREAD ¯1 82 10000000 ((J-1)×10000000)
ŒNUNTIE ¯1
ŒTCLF
'SEGMENT ',J,' READ'
```

```
–((10000000+DIM)>½GTF)/'FLAG„1 ª ŒTCLF ª ''FLAG SET TO 1'''

 © NEWGTF2„NEWGTF„0 82½' '
 IND„(GTF=ŒTCLF)/¼½GTF
 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•(½GTF)÷1
000000)'
 IND„(GTF=ŒTCLF)/¼½GTF
 …(0=½IND)/0
 LINE„IND[1]†GTF
 LINE„¯1‡LINE
 IND2„(LINE=ŒAV[10])/¼½LINE
COL3„¯1‡(IND2[2]‡(IND2[3]†LINE))
 TEST„^/COL3[¼3]='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†(COL3,15½' ')
COL9A„40†(COL9,40½' ')
COL4A„10†(COL4,10½' ')
COL5A„10†(COL5,10½' ')
NEWLINE„COL1,'  ',COL2,'  ',COL3A,'  ',COL4A,'  ',COL5A,'  ',COL6,'
 ', COL7,'  ',COL8,'  '
NEWGTF2„NEWGTF2,[1]NEWLINE

DOWN:IND„(GTF=ŒTCLF)/¼½GTF
GTF„IND[1]‡GTF
…((FLAG=0)^(0=½IND))/RETJ
…RETI

© ‘LACCDS„NEWGTF2

ŒTCLF
'PROGRAM READLACGFF COMPLETE.  CDS DATA IN VARIABLE ‘LACCDS'
'THIS VARIABLE NEEDS TO BE MANUALLY REORDERED
```

****

```
CONVLAC
© THIS PROGRAM CONVERTS THE CHARACTER VARIABLE ‘LACCDS2 TO THE NUMER
IC VARIABLE ‘LACCDS3
©  SORTS IT AND ELIMINATES DUPLICATE ENTRIES, PRODUCING THE VARIABLE
 ‘LACCDS4
‘LACCDS3„0 5½0
DIM„1†½‘LACCDS2
ŒTCLF
DIM
START„0

I„0
RETI:I„I+1
TEST„(I÷1000)=(—(I÷1000))
—(TEST=1)/'I'
LINE„‘LACCDS2[I;]
CONTIG„ŒFI LINE[3+¼8]
STARTPOS„ŒFI LINE[49+¼11 ]
ENDPOS„ŒFI LINE[61+¼12]

STRAND„0
—(LINE[77]='+')/'STRAND„1'
PHASE„ŒFI LINE[80]
NEWLINE„CONTIG,STARTPOS,ENDPOS,STRAND,PHASE
‘LACCDS3„‘LACCDS3,[1]NEWLINE
…(I<DIM)/RETI

ŒTCLF
'STARTING CONVLAC2'
CONVLAC2
ŒTCLF
'STARTING CONVLAC3'
CONVTRIF3
‘LACCDS4„‘LACCDS3[INDEX9;]
ŒTCLF
'PROGRAM FINISHED.  DATA ON I. LACUNOSA CODING SEQUENCES IN VARIABLE
 ''‘LACCDS4''.'
 ****************************
FIXLACCDS4
©  THIS PROGRAM ELIMINATES FURTHER DUPLICATES IN ‘LACCDS4 AND OVERLA
PPING CDS FEATURES

‘LACCDS5„0 5½0

DIM„½‘UNIQSCAFFS
DIM1„1†½‘LACCDS4

© I LOOP LOOPS THROUGH SCAFFOLDS
I„0
RETI:I„I+1
TEST„(I÷10)=(—(I÷10))
—(TEST)/'I'
SCAFF„‘UNIQSCAFFS[I]
IND„(‘LACCDS4[;1]=SCAFF)/¼DIM1
PART„‘LACCDS4[IND;]
```

```
PART„PART[“PART[;2];]
DIMPART„1†½PART
TEST„1=DIMPART
–(TEST)/''‘LACCDS5„‘LACCDS5,[1]PART ª …DOWN2'

© J LOOP REMOVES CDS FEATURES WITH DUPLICATE BEGINNING POSITIONS
DUPS„0½0
DIM2„1†½PART
KEEP„0 5½0
J„0
RETJ:J„J+1

LINE1„,PART[J;]
LINE2„,PART[J+1;]

–(LINE1[2]¬LINE2[2])/'KEEP„KEEP,[1]LINE1 ª …DOWN1'
DUPS„DUPS,J
IND„(PART[;2]=LINE1[2])/¼DIM2

NEQUAL„½IND
LINES„PART[IND;]

IND2„1†((–/LINES[;3])=LINES[;3])/¼½IND
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„0½0
DIM3„1†½KEEP

–(DIM3=1)/''‘LACCDS5„‘LACCDS5,[1]KEEP ª …DOWN2'
KEEP2„0 5½0
K„0
RETK:K„K+1

LINE1„,KEEP[K;]
LINE2„,KEEP[K+1;]

–(LINE1[3]¬LINE2[3])/'KEEP2„KEEP2,[1]LINE1 ª …DOWN3'
DUPS2„DUPS,K
IND„(KEEP[;3]=LINE1[3])/¼DIM3

NEQUAL„½IND
LINES„KEEP[IND;]

IND2„1†((–/LINES[;3])=LINES[;3])/¼½IND
ADDLINE„LINES[IND2;]
KEEP2„KEEP2,[1]ADDLINE
```

```
K„K+(NEQUAL-1)
DOWN3:…(K<DIM3-1)/RETK

‘LACCDS5„‘LACCDS5,[1]KEEP2

DOWN2:…(I<DIM)/RETI
*****************************
INDEXLAC ;STARTBYTE
© THIS PROGRAM INDEXES THE SCAFFOLDS OF THE I. LACUNOSA GENOME.  IT
CREATES THE VARIABLE
©   ‘LACINDEX, 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' ŒNTIE ¯1
PARTSCAFF„ŒNREAD ¯1 82 2000000 STARTBYTE
ŒNUNTIE ¯1

DIM„½PARTSCAFF
…(DIM=0)/0

IND„(PARTSCAFF ='>')/¼½PARTSCAFF
ENDSCAFF„IND[2]-1

TEMPSCAFF„ENDSCAFF†PARTSCAFF
IND„(TEMPSCAFF=ŒTCLF)/¼½TEMPSCAFF
HEAD„IND[1]†TEMPSCAFF
SEQ„IND[1]‡TEMPSCAFF
SEQ„(SEQ¬ŒTCLF)/SEQ
‘SCAFF„ŒFI HEAD[4+¼8]

SEQ[¼10]
READLACSCAFF ‘SCAFF
‘SEQUENCE[¼10]
TRSH„Œ

©  ‘LACINDEX„‘LACINDEX,[1](‘SCAFF,(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 ‘LACCDS5 TO FORM ‘LACSNPS, 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 ‘SNPNUM (SNP NUMBER, AS COUNTED BY THIS PROGRAM)

‘LACSNPS„0 10½0
‘LACCODONS„0 19½' '   © THIS VARIABLE CONTAINS CODON INFORMATION FOR
 THE CORRESPONDING SHP
                      ©    COMUMN1  ALTERNATIVE ALLELES
                      ©    COLUMN2: ALTERNATIVE CODON 1
                      ©    COLUMN3: ALTERNATIVE CODON 2
                      ©    COLUMN4: SYNONYMOUS(S) OR NON-SYNONYMOUS
(N) DIFFERENCE
‘TSCORES„0 62 2½' '   ©  THIS VARIABLE CONTAINS ALL OF THE SNP CALLS
 IN SAME ORDER AS ‘LACSNPS AND ‘LACCODONS
PROBLEMS„0 3½0
NOTFOUND„0½0
‘SNPINFO„0 3½0
‘ALTALLELES„0 2½' '
‘UNEQUAL2„0

'C:\JG1' ŒNTIE ¯1
SIZE„ŒNSIZE ¯1
FILE„ŒNREAD (¯1,82,(SIZE+1000),0)   ©  READ IN FILE
ŒNUNTIE ¯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

© ‘SNPS„0 4½0    © 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'

‘SNPNUM„I      © SNIP NUMBER

…(‘SNPNUM>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 ŒSS 'tig'
TEMP3„TEMP2/¼½TEMP2
TEMP4„(TEMP3[2]-1)†TEMP
TEMP5„(TEMP3[1]-1)‡TEMP4
INDA„(TEMP5=ŒAV[10])/¼½TEMP5
TEMP7„TEMP5[3+¼8]
‘SCAFF„ŒFI 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='/')/¼½SCORES
SCORES[IND]„' '
IND„(~SCORES¹(' ',ŒAV[10]))/¼½SCORES
SCORES2„SCORES[IND]
SCORES3„62 2½SCORES2

NONAUSTSCORES„,SCORES3[‘NONAUSTINDEX;]      © TEST FOR VARIATION BESI
DES AUSTINII
NUCS„'ACGT*'

TEST1„+/NUCS¹NONAUSTSCORES
—(TEST1=1)/'  …DOWN4'    ©  IF NO VARIATION AFTER REMOVE AUSTINII, SK
IP SNP

TEMP8„¯1‡INDA[1]‡INDA[2]†TEMP5
‘SCAFFPOS„ŒFI TEMP8               ©  SCAFFOLD POSITION OF SNP

© DETERMINE WHICH ELEMENT OF ‘LACCDS5 IS RELEVANT
IND1„‘LACCDS5[;1]=‘SCAFF          © BOOLEAN, 1 IF SCAFFOLD
IND2„‘LACCDS5[;2]ˆ‘SCAFFPOS       © BOOLEAN, 1 IF SNP POSITION ‰ STAR
T POSITION OF CDS FEATURE
IND3„‘LACCDS5[;3]‰‘SCAFFPOS       © BOOLEAN, 1 IF SNP POSITION ˆ END
POSITION OF CDS FEATURE
```

```
IND„(IND1^IND2^IND3)/¼½IND1        © INDEX OF ENTRY THAT SATISFIES EA
CH OF ABOVE CONDITIONS
FLAG„½IND
…(FLAG¬0)/ DOWN20                 © IF CDS FOUND, GO TO DOWN20
©IF NO CDS FOUND, PROCESS SNP

NEWLINE„‘SCAFF,¯1,‘SCAFFPOS,¯1,¯1,¯1,¯1,¯1,¯1,‘SNPNUM
‘LACSNPS„‘LACSNPS,[1](NEWLINE)     © ADD NEW LINE TO ‘LACSNPS

CODON1„'XXX'
CODON2„'XXX'
STATUS„'0'                    © THIS STATUS INDICATES NO CDS FOUND
CLINE„'    ',CODON1,'  ',CODON2,'  ',STATUS,7 0•‘SNPNUM      © MAKE LINE
 LISTING CODONS AND STATUS (SYN VS. NON-SYN)
‘LACCODONS„‘LACCODONS,[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„,‘LACCDS5[IND4;]             © PICK APPROPRIATE LINE FROM ‘LACCD
S5
STARTPOS„LINE[2]
END„LINE[3]
STRAND„LINE[4]
PHASE„LINE[5]

© NEXT, PICK OUT CODON CORRESPONDING TO SNP

READLACSCAFF ‘SCAFF         © READ IN SCAFFOLD SEQUENCE, IN VARIABLE ‘
SEQUENCE
© DIFF„‘SCAFFPOS-STARTPOS     © DIFFERENCE BETWEEN SNP POSITION AND S
TART POSITION OF CDS FEATURE

–(CALLEDBASE¬‘SEQUENCE[‘SCAFFPOS])/'‘UNEQUAL2„‘UNEQUAL2+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„‘SCAFFPOS-STARTFRAME     © NUCLEOTIDES BETWEEN STARTFRAME AND SC
AFFOLD POSITION
POSFRAME„3|DIFF               © CODON POSITION OF SNP
STCOD„‘SCAFFPOS-POSFRAME      © START POSITION OF CODON CONTAINING TH
E SNP
CODONINDEX„STCOD,(STCOD+1),(STCOD+2)    © POSITIONS OF ALL THREE NUCS
 OF CODON
CODON„‘SEQUENCE[CODONINDEX]    ©  CODON EXTRACTED FROM I. LACIDA SEQU
ENCE
CODPOS„(‘SCAFFPOS=CODONINDEX)/¼3    © VARIABLE POSITION IN CODON
```

```
©  ADD INFORMATION TO ‘LACSNPS
NEWLINE„‘SCAFF,STARTPOS,‘SCAFFPOS,END,STRAND,PHASE,CODONINDEX,‘SNPNU
M
‘LACSNPS„‘LACSNPS,[1]NEWLINE

 TEST„',''¹ALLELE2    © 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•‘SNPNUM     © MAKE LINE
 LISTING CODONS AND STATUS (SYN VS. NON-SYN)
‘LACCONDONS„‘LACCODONS,[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 ª 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•‘SNPNUM     ©
MAKE LINE LISTING CODONS AND STATUS (SYN VS. NON-SYN)
‘LACCODONS„‘LACCODONS,[1]CLINE          © ADD LINE TO ‘LACCODONS

DOWN4:DATA2„(TEMP3[2]-1)‡DATA2       ©  DROP CURRENT SNP FROM DATA2

TEST„+/(DATA2 ŒSS 'tig')

…(TEST>1)/RETI
ŒTCLF
'PROGRAM COMPLETE.  SNP DATA IN VARIABLES '''‘LACSNPS'' AND '''‘LACCOD
ONS''.'

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

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

STATUS„‘LACCODONS[;12]      © COLUMN OF N'S, S'S, O'S
SNPNUMS„‘LACCODONS[;12+¼7]     © SNP NUMBERS FROM ‘LACCODONS IN TEXT
FORMAT
BLANKS„((½STATUS)½' ')         ©   ADD ONE COLUMN OF SPACES
SNPNUMS„,(SNPNUMS,BLANKS)
SNPNUMS2„ŒFI SNPNUMS           © CHANGE SNP NUMBERS TO NUMERIC

© GET SYN SNPS

IND1„(STATUS='S')/¼½STATUS     © INDEX OF WHICH ROWS OF ‘LACODONS COR
RESPOND TO SYNONYMOUS SNPS

SYNSNPNUMS„SNPNUMS2[IND1]      © PICK SNP NUMBERS CORRESPONDING TO SY
NONYMOUS SITES

DIMS„½‘SNPS1
‘SYNSCORES„0 62 2½0
I„0
RETI:I„I+1

TEST„(I÷1000)=(—(I÷1000))
—(TEST)/'I'
TEST„‘SNPS1[I]¹SYNSNPNUMS
—(TEST)/'‘SYNSCORES„‘SYNSCORES,[1]‘SCORES[I;;]'
…(I<DIMS)/RETI

© GET NON-SYN SNPS
'PROCESSING NON-SYN SNPS'

IND2„(STATUS='N')/¼½STATUS
NONSNPNUMS„SNPNUMS2[IND2]

DIMN„½‘SNPS1
‘NONSCORES„0 62 2½0
J„0
RETJ:J„J+1

TEST„(J÷1000)=(—(J÷1000))
—(TEST)/'J'
TEST„‘SNPS1[J]¹NONSNPNUMS
—(TEST)/'‘NONSCORES„‘NONSCORES,[1]‘SCORES[J;;]'
…(J<DIMN)/RETJ

DOWN:'PROCESSING O SNPS'
IND3„(STATUS='O')/¼½STATUS

OSNPNUMS„SNPNUMS2[IND3]

DIMO„½‘SNPS1
‘OSCORES„0 62 2½0
K„0
```

```
RETK:K„K+1

TEST„(K÷1000)=(–(K÷1000))
–(TEST)/'K'
TEST„‘SNPS1[K]¹OSNPNUMS
–(TEST)/'‘OSCORES„‘OSCORES,[1]‘SCORES[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

‘TOTSYN„‘TOTNONSYN„0
DIMTRAN„1†½‘LACTRANDATA

LACSCAFFS„‘LACINDEX[;1]

COUNTER„1    © THIS IS A COUNTER FOR TRANSCRIPT NUMBER

‘LACTRANIND„0½' '      ©  THIS IS THE VARIABLE THAT WILL HOLD SUCCES
SIVE  SPLICED TRANSCRIPT INDICIES

                      © FORMAT IS  > TRANSCRIPTNUMBER SCAFFNUM POSI
TIONINDEX ŒTCLF
                      © 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

‘LACTRANINDDATA„0 8½0   © 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 ‘L
ACTRANIND,END POSITION OF TRANSCRIPT DATA IN ‘LACTRANIND

LACTRANLENGTH„0

I„0
RETI:I„I+1
TEST„(I÷100)=(–(I÷100))
–(TEST=1)/'I'

LINE1„‘LACTRANDATA[I;]     © READ IN DATA FOR TRANSCRIPT I
SCAFF„LINE1[2]           © LACUNOSA SCAFFOLD CORRESPONDING TO TRANSCR
IPT

–(~SCAFF¹LACSCAFFS)/'ŒTCLF ª ''NO CONTIG FOUND'' ª TRSH„Œ ª …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]ˆLINE1[4]     © DITTO
IND„IND1^IND2^IND3

–(0=+/IND)/' …DOWN'                 © SKIP IF TRANSCRIPT CONTAINS NO
I. LACCDS FEATURES

IND„IND/¼½IND

PARTCDS4„‘LACCDS5[IND;]        © PICK I. LAC CDS FEATURES CONTAINED I
N TRANSCRIPT
DIMPART„1†½PARTCDS4
IND1„(PARTCDS4[;4]=1)/¼DIMPART     © INDEX FOR CDS ON POSITIVE STRAND
IND2„(PARTCDS4[;4]=0)/¼DIMPART     © INDEX FOR CDS ON NEGATIVE STRAND

PARTPOS„PARTCDS4[IND1;]            © PICK CDS ON POSITIVE STRAND

…(0=1†½PARTPOS)/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„0½' '
DIM1„1†½PARTPOS
…(DIM1=0)/DOWN10

© START POSITIVE STRAND LOOP

CDSIND2„CDSIND„0½0

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)+¼(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),ŒTCLF

   ‘LACTRANIND„‘LACTRANIND,LINE5

   LACTRANSTART„LACTRANLENGTH+1
   LACTRANLENGTH„LACTRANEND„LACTRANLENGTH+½LINE5

   LINE6„COUNTER,SCAFF,STRAND,BEGPHASE,(1†CDSIND),(¯1†CDSIND),LACTRA
NSTART,LACTRANEND
   ‘LACTRANINDDATA„‘LACTRANINDDATA,[1]LINE6
   COUNTER„COUNTER+1


COUNTSYN SEQ       © CALL SCRIPT COUNTSYN
‘TOTSYN„‘TOTSYN+SYN
‘TOTNONSYN„‘TOTNONSYN+NONSYN

…(0=1†½PARTNEG)/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„0½' '
DIM1„1†½PARTNEG
…(DIM1=0)/DOWN
CDSIND2„CDSIND„0½0
J„0
RETJ:J„J+1
LINE2„PARTNEG[J;]
PHASE„LINE2[5]

CDSSTARTPOS„LINE2[2]
CDSENDPOS„LINE2[3]
INDB„(CDSSTARTPOS-1)+¼(1+(CDSENDPOS-CDSSTARTPOS))
CDSIND„CDSIND,CDSSTARTPOS,CDSENDPOS
CDSIND2„CDSIND2,((CDSSTARTPOS-1)+¼(1+(CDSENDPOS-CDSSTARTPOS)))
```

```
DOWN2:…(J<DIM1)/RETJ

 BEGPHASE„PARTNEG[1;5]

SPLICETRAN2„‘SEQUENCE[CDSIND2]
SPLICETRAN„REVCOMP SPLICETRAN2
TRANSLATE SPLICETRAN

COUNT1„+/AA1='*'
COUNT2„+/AA2='*'
COUNT3„+/AA3='*'


COUNTS„COUNT1,COUNT2,COUNT3
MIN„˜/COUNTS
INDC„(COUNTS=MIN)/¼3
–((½INDC)>1)/'…DOWN'

CDSIND2„(¯1×(INDC-1))‡CDSIND2

STRAND„0
LINE5„'> ',(8 0•COUNTER),(8 0•SCAFF),(10 0•CDSIND),ŒTCLF
‘LACTRANIND„‘LACTRANIND,LINE5

LACTRANSTART„LACTRANLENGTH+1
LACTRANLENGTH„LACTRANEND„LACTRANLENGTH+½LINE5

LINE6„COUNTER,SCAFF,STRAND,BEGPHASE,(1†CDSIND),(¯1†CDSIND),LACTRANST
ART,LACTRANEND

‘LACTRANINDDATA„‘LACTRANINDDATA,[1]LINE6
COUNTER„COUNTER+1

SEQ„‘SEQUENCE[CDSIND2]
SEQ2„REVCOMP SEQ
COUNTSYN SEQ2
‘TOTSYN„‘TOTSYN+SYN
‘TOTNONSYN„‘TOTNONSYN+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„½X
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„(½SEQ)÷3
MAT„(LENGTH2,3)½SEQ
MAT„³MAT

TEMP„+/(‘CODONS^.=MAT)
TEMP„((½TEMP),1)½TEMP


TEMP2„TEMP,TEMP,TEMP

TEMP3„‘DEGENMATRIX×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„˜(½X)÷3
CODONS„³(MAX,3)½X
FIXFN
AA„,(1 65½¼65)+.×(‘CODE^.=CODONS)
AA1„,‘SYMB[AA;]
X„1‡X
MAX„˜(½X)÷3
CODONS„³((MAX,3)½X)
FIXFN
AA„,(1 65½¼65)+.×(‘CODE^.=CODONS)
AA2„,‘SYMB[AA;]
X„1‡X
MAX„˜(½X)÷3
CODONS„³((MAX,3)½X)
FIXFN
AA„,(1 65½¼65)+.×(‘CODE^.=CODONS)
AA3„,‘SYMB[AA;]
****************************
```

The script TRANSLATE requires the variables `‘CODE` (a 65 x 3 character matrix) and `‘SYMB` (a 65 x 1 character vector):

Transpose of `‘CODE =`

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

Transpose of `‘SYMB =`

```
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
©   ‘LACINDEX, 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

‘LACINDEX„0 3½0

STARTBYTE„0
I„0
RETI:I„I+1

TEST„(I÷100)=(—(I÷100))
—(TEST=1)/'I'

'C:\LACGENOME' ŒNTIE ¯1  © OPEN LAC GENOME FASTA FILE

PARTSCAFF„ŒNREAD ¯1 82 5000000 STARTBYTE
ŒNUNTIE ¯1

DIM„½PARTSCAFF
…(DIM=0)/0

IND„(PARTSCAFF ='>')/¼½PARTSCAFF
ENDSCAFF„IND[2]-1

TEMPSCAFF„ENDSCAFF†PARTSCAFF
IND„(TEMPSCAFF=ŒTCLF)/¼½TEMPSCAFF
HEAD„IND[1]†TEMPSCAFF
SEQ„IND[1]‡TEMPSCAFF
SEQ„(SEQ¬ŒTCLF)/SEQ
‘SCAFF„ŒFI HEAD[17+¼5]

‘LACINDEX„‘LACINDEX,[1](‘SCAFF,(STARTBYTE),(STARTBYTE+ENDSCAFF))
STARTBYTE„STARTBYTE+ENDSCAFF

…RETI
*********************
```

```
READLACSCAFF    SCAFFN

© BEFORE RUNNING THIS, MAKE SURE TO RUN 'CONVLACINDEX' TO CONVERT ‘L
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„‘LACINDEX[;1]
IND„(SCAFFN=SCAFFNUMS)/¼½SCAFFNUMS
TEMP1„,‘LACINDEX[IND;]
STARTBYTE„TEMP1[2]
ENDBYTE„TEMP1[3]

'C:\LACGENOME' ŒNTIE ¯1
 SCAFFOLD„ŒNREAD ¯1 82 (ENDBYTE-STARTBYTE) STARTBYTE
ŒNUNTIE ¯1
SCAFFNAME„24†SCAFFOLD
‘SEQUENCE„24‡SCAFFOLD
‘SEQUENCE„(‘SEQUENCE¬ŒTCLF)/‘SEQUENCE
******************************
```

READLACSCAFF requires running ‘CONVLACINDEX’ before running:

```
******************************
CONVLACFINDEX
© THIS PROGRAM CONVERTS ‘LACINDEX, A CHARACTER MATRIX, TO ‘LACINDEX,
©    A NUMERIC MATRIX


DIM„1†½‘LACINDEX
TEMP1„‘LACINDEX,(DIM,3)½' '
TEMP2„,TEMP1
IND„(TEMP2=ŒAV[10])/¼½TEMP2
TEMP2[IND]„' '
TEMP3„ŒFI TEMP2
‘LACINDEX„(DIM,3)½TEMP3
******************************
```

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 ‘PI2 THAT HAS THE AVERAGE PAIRWISE PI VALUES
FOR EACH PAIR OF SAMPLES
©  PROGRAM READS IN VARIABLE ‘SCORES CREATED BY READCALLS AS X
©    (CAN ALSO READ IN ‘SYNSCORES OR ‘NONSCORES OR ‘RSCORES)

© DIVIDER„‘TOTSYN   ©  CHANGE THIS IF CALCULATING PI FOR NON-SYN SIT
ES
© DIVIDER„‘TOTNONSYN  © FOR USE WITH ‘RSCORES
DIVIDER„30036768   ©  TOTAL NUMBER OF SITES IN TRANSCRIPTOME

DIM„1†½X

PICUM„62 62½0   © SET UP PI MATRIX- WILL EVENTUALLY HAVE NUMBER OF P
AIRWISE DIFFERENCES ACROSS ALL VARIABLE SNPS
PICOUNT„62 62½0   © 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)/'ŒTCLF ª 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 ‘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]='.')/¼62
COUNTMAT„62 62½1
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
‘PI2„PICUM×CORRCOUNT÷DIVIDER
ŒTCLF
'PROGRAM PICALC2 FINISHED.'
'PAIRWISE PI VALUES IN VARIABLE ''‘PI2''.'
******************************
```

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 ''‘PI2', WHICH IS PICUM÷‘TOTNUC, WHERE PICUM I
S INTERMEDIATE
©   MATRIX FROM 'PICALC2', AND ‘TOTNUC IS TOTAL NUMBER OF SITES IN T
RANSCRPTOMES, AS CALCULATED BY 'SPLICE'

CINDEX„‘CINDEX      © INDEX FOR WHICH CORDATATRILOBA SAMPLES BEING US
ED
LINDEX„‘LINDEX      © 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„½CINDEX
MASKC„(TMP,TMP)½1
I„0
RETI:I„I+1
MASKC[I;I]„0
…(I<TMP)/RETI

TMP„½LINDEX
MASKL„(TMP,TMP)½1
J„0
RETJ:J„J+1
MASKL[J;J]„0
…(J<TMP)/½RETJ

© CALCULATE WITHIN-CORDAT AVERAGE PI
CW1„‘PI2[CINDEX;CINDEX]     © CHOOSE SUBSET OF ‘PI2 CORRESPONDING TO
CORDAT
CW2„CW1×MASKC              © MAKE DIAGONAL ELEMENTS 0
DIM1„½CINDEX                © 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„½LINDEX
LW3„(+/+/LW2)÷(DIM2×(DIM2-1))

© CALCULATE BETWEEN-SPECIES AVERAGE PI
B„‘PI2[CINDEX;LINDEX]
B2„(+/+/B)÷(DIM1×DIM2)

© COMMENT NEXT STATEMENTS IF RUNNING PIBOOT4


ŒTCLF
'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 2½0    ©  VARIABLE TO HOLD BOOTSTRAP DATA

MASK4„62 62½1

DIM„1†½‘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„½CINDEX
DIML„½LINDEX
©  BOOTSTRAP CORDAT AND LAC SAMPLES
RANDC„?DIMC½DIMC
SCORES[;CINDEX;]„SCORES[;CINDEX[RANDC];]
```

```
RANDL„?DIML½DIML
SCORES[;LINDEX;]„SCORES[;LINDEX[RANDL];]

© NOW BOOTSTRAP SNPS
DIMSNPS„1†½SCORES
RANDSNPS„?DIMSNPS½DIMSNPS
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„+/DIFFˆ0
ŒTCLF
'PROPORTION OF 1000 BOOTSTRAP SAMPLE DIFFS ˆ0: ',PROP

CVALS„,BOOTDATA2[;1]
CVALS2„CVALS[“CVALS]
LVALS„,BOOTDATA2[;2]
LVALS2„LVALS[“LVALS]
CONFC„CVALS2[25,975]
CONFL„LVALS2[25,975]
ŒTCLF
'CONF INTERVAL FOR I. CORD: ',CONFC
'CONF INTERVAL FOR I. LAC:  ',CONFL
ŒTCLF
'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 ‘PI2 MATRIX CALCULATED BY 'PICALC2'

MASK4„62 62½1

I„0
RETI:I„I+1
MASK4[I;I]„0
…(I<62)/RETI

PI„MASK4×‘PI2
```

```
PIBETWALLO„PI[‘CALLO3;‘LALLO3]
PIBETWALLOCLOSE„PI[‘CALLOCLOSE;‘LALLOCLOSE]
PIBETWSYMP„PI[‘CSYMPINDEX;‘LSYMPINDEX]
PICBETWALLOSYMP„PI[‘CALLO3;‘CSYMPINDEX]
PILBETWALLOSYMP„PI[‘LALLO3;‘LSYMPINDEX]
PICBETWALLOCLOSESYMP„PI[‘CALLOCLOSE;‘CSYMPINDEX]
PILBETWALLOCLOSESYMP„PI[‘LALLOCLOSE;‘LSYMPINDEX]


AVEBETWSYMP„(+/+/PIBETWSYMP)÷((½‘CSYMPINDEX)×((½‘LSYMPINDEX)))
AVEBETWALLO„(+/+/PIBETWALLO)÷((½‘CALLO3)×((½‘LALLO3)))
AVEBETWALLOCLOSE„(+/+/PIBETWALLOCLOSE)÷((½‘CALLOCLOSE)×((½‘LALLOCLOS
E)))
AVECBETWALLOSYMP„(+/+/PICBETWALLOSYMP)÷((½‘CALLO3)×(½‘CSYMPINDEX))
AVELBETWALLOSYMP„(+/+/PILBETWALLOSYMP)÷((½‘LALLO3)×(½‘LSYMPINDEX))

AVECBETWALLOCLOSESYMP„(+/+/PICBETWALLOCLOSESYMP)÷((½‘CALLOCLOSE)×(½‘
CSYMPINDEX))
AVELBETWALLOCLOSESYMP„(+/+/PILBETWALLOCLOSESYMP)÷((½‘LALLOCLOSE)×(½‘
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

ŒTCLF
'BETWEEN SPECIES COMPARISIONS'
ŒTCLF
'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)
ŒTCLF
'WITHIN SPECIES COMPARISONS'
ŒTCLF
'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)


ŒTCLF
'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 11½0      © INITIALIZE MARIX TO HOLD OUTPUT DATA

MASK4„62 62½1

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„?(½COMBCINDEX)½(½COMBCINDEX)
LIND„?(½COMBLINDEX)½(½COMBLINDEX)

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„?(½COMBCINDEX2)½(½COMBCINDEX2)
LIND„?(½COMBLINDEX2)½(½COMBLINDEX2)

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

ŒTCLF
'PROPORTION OF 1000 BOOTSTRAP SAMPLES ‰ OBSERVED PI CONTRASTS'
'   FIRST NUMBER IS OBSERVED, SECOND IS  PROPORTION'
ŒTCLF
'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

ŒTCLF
'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

ŒTCLF
'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„½‘CSYMPINDEX
N2„½‘LSYMPINDEX
N3„½‘CALLO3
N4„½‘LALLO3
N5„½‘CALLOCLOSE
N6„½‘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

‘DIFFS„0 5½0
‘DIFFS2„0 5½0
DIM„1†½‘SCORES
I„0
RETI:I„I+1
TEST„(I÷1000)=(˜(I÷1000))
–(TEST)/'I'

SCORES„‘SCORES[I;;]
IND„¼61
IND2„(IND¬27)/IND
SCORES2„SCORES[IND;]
IND„(SCORES2[;1]¬'.')/¼1†½SCORES2
SCORES3„SCORES2[IND;]
COUNTS„,(+/[1]((,SCORES3)°.=‘ALLELES))
MAX„–/COUNTS
IND„(COUNTS=MAX)/¼5
MAXALLELE„'CTGA*'[IND]
MAXALLELE„1†MAXALLELE

CASCORES„,SCORES[‘CALLO3;]
CSSCORES„,SCORES[‘CSYMPINDEX;]
LASCORES„,SCORES[‘LALLO3;]
LSSCORES„,SCORES[‘LSYMPINDEX;]

FCA„(+/CASCORES=MAXALLELE)÷(½CASCORES)
FCS„(+/CSSCORES=MAXALLELE)÷(½CSSCORES)
FLA„(+/LASCORES=MAXALLELE)÷(½LASCORES)
FLS„(+/LSSCORES=MAXALLELE)÷(½LSSCORES)
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
‘DIFFS„‘DIFFS,[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
‘DIFFS2„‘DIFFS2,[1](CDIFF,LDIFF,ADJALLOMID,ADJSYMPMID,MIDDIFF)

DOWN:…(I<DIM)/RETI

SUM„+/[1]‘DIFFS
AVES„SUM÷(1†½‘DIFFS)
ŒTCLF
'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]‘DIFFS2
AVES2„SUM2÷(1†½‘DIFFS2)
ŒTCLF
'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 ‘DIFFS

ŒTCLF
'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
‘AVES„0 5½0

J„0
RETJ:J„J+1
'J= ',J

‘SCORES2„‘SCORES
IND1„?28½28
‘SCORES2[;‘CINDEX3;]„‘SCORES2[;‘CINDEX3[IND1];]
IND2„?28½28
‘SCORES2[;‘LINDEX3;]„‘SCORES2[;‘LINDEX3[IND2];]


‘DIFFS„0 5½0
‘DIFFS2„0 5½0
DIM„1†½‘SCORES2
I„0
RETI:I„I+1
TEST„(I÷10000)=(˜(I÷10000))
—(TEST)/'I'

SCORES„‘SCORES2[I;;]
IND„¼61
IND2„(IND¬27)/IND
SCORES2„SCORES[IND;]
IND„(SCORES2[;1]¬'.')/¼1†½SCORES2
SCORES3„SCORES2[IND;]
COUNTS„,(+/[1]((,SCORES3)°.=‘ALLELES))
MAX„—/COUNTS
IND„(COUNTS=MAX)/¼5
MAXALLELE„'CTGA*'[IND]
MAXALLELE„1†MAXALLELE

CASCORES„,SCORES[‘CALLO3;]
CSSCORES„,SCORES[‘CSYMPINDEX;]
LASCORES„,SCORES[‘LALLO3;]
LSSCORES„,SCORES[‘LSYMPINDEX;]

FCA„(+/CASCORES=MAXALLELE)÷(½CASCORES)
```

```
FCS„(+/CSSCORES=MAXALLELE)÷(½CSSCORES)
FLA„(+/LASCORES=MAXALLELE)÷(½LASCORES)
FLS„(+/LSSCORES=MAXALLELE)÷(½LSSCORES)
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
‘DIFFS„‘DIFFS,[1](ALLODIFF,SYMPDIFF,DIFF,CDIFF1,LDIFF1)

DOWN:…(I<DIM)/RETI

SUM„+/[1]‘DIFFS
AVES„SUM÷(1†½‘DIFFS)
‘AVES„‘AVES,[1]AVES
ŒTCLF
'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„‘AVES[;1]
QQ„Q[“Q]
ŒTCLF
'95 PERCENT CONF INTERVAL FOR ALLOPATRIC FREQ DIFF BETWEEN SPECIES:'
'          (',(6 3•QQ[LOW]),', ',(6 3•QQ[UP]),')'

Q„‘AVES[;2]
QQ„Q[“Q]
ŒTCLF
'95 PERCENT CONF INTERVAL FOR SYMPATRIC FREQ DIFF BETWEEN SPECIES:'
'          (',(6 3•QQ[LOW]),', ',(6 3•QQ[UP]),')'

Q„‘AVES[;3]
QQ„Q[“Q]
ŒTCLF
'95 PERCENT CONF INTERVAL FOR ALLO - SYM FREQ DIFF BETWEEN SPECIES:'
'          (',(6 3•QQ[LOW]),', ',(6 3•QQ[UP]),')'

Q„‘AVES[;4]
QQ„Q[“Q]
ŒTCLF
```

```
'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]
ŒTCLF
'95 PERCENT CONF INTERVAL FOR LAC ALLO-SYM FREQ DIFF BETWEEN SPECIES
:'
'          (',(6 3•QQ[LOW]),', ',(6 3•QQ[UP]),')'


ŒTCLF
'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 2½1 1.1 .9 1 .8 .9 .7 .8 .6 .7 .5 .6 .4 .5 .3 .4 .2 .3 .1
  .2 0 .1
‘OUTDATA„0 5½0
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 1½‘OUTDATA[;5]
ŒTCLF
'PIN/PIS VALUES FOR DIFFERENT FREQUENCY-DIFFERENCE BINS'
TEMP„CUTOFFS[¼10;],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†½‘SYNSCORES
NUCS„'ACGT*'
‘FIXEDSNPS„0½0     © THIS VARIABLE CONTAINS SNP NUMBERS OF SNPS SHOWI
NG FIXED DIFFS BETWEEN SPECIES
‘FIXEDSNPPROPS„0 14½' '      ©  THIS IS A CHARACTER MATRIX WITH INFOR
MATION ON FIXED SNPS

CINDEX„‘CINDEX          © USE ‘CINDEX FOR ALL SAMPLES, ‘CALL03 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 ¹'ACGT')/CSCORES1
LSCORES„(LSCORES1¹'ACGT')/LSCORES1

ALLELEIND„NUCS¹(CSCORES,LSCORES)

ALLELES„ALLELEIND/NUCS
—(1=½ALLELES)/' FIXEDSYN„FIXEDSYN+1 ª …DOWN1'

ALLELE1„ALLELES[1]
ALLELE2„ALLELES[2]
FREQ1„(+/(CSCORES=ALLELE1))÷(½CSCORES)
FREQ2„(+/(LSCORES=ALLELE1))÷(½LSCORES)
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†½‘NONSCORES
NONCOUNT„0 0
J„0
RETJ:J„J+1
SCORES„‘NONSCORES[J;;]
CSCORES„,SCORES[CINDEX;]
LSCORES„,SCORES[LINDEX;]

CSCORES„(CSCORES ¹'ACGT')/CSCORES
LSCORES„(LSCORES¹'ACGT')/LSCORES

ALLELEIND„NUCS¹(CSCORES,LSCORES)
ALLELES„ALLELEIND/NUCS

ALLELES„ALLELEIND/NUCS
—(1=½ALLELES)/'FIXEDNON„FIXEDNON+1 ª …DOWN2'

ALLELE1„ALLELES[1]
ALLELE2„ALLELES[2]
FREQ1„(+/(CSCORES=ALLELE1))÷(½CSCORES)
FREQ2„(+/(LSCORES=ALLELE1))÷(½LSCORES)
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†½‘SYNSCORES)-((+/SYNCOUNT)+FIXEDSYN)
NONLOWER„(1†½‘NONSCORES)-((+/NONCOUNT)+FIXEDNON)
ŒTCLF
'M-K TABLE FOR CUTOFF ',CUTOFF
ŒTCLF
'           SYN      NONSYN'

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

ALPHA„1-(SYNFOCUS×NONLOWER)÷(NONFOCUS×SYNLOWER)
NUMBER„ALPHA×NONFOCUS
ŒTCLF
'ALPHA, NUMBER ',ALPHA,NUMBER
ŒTCLF
MATRIX„2 2½SYNFOCUS,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†½‘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„(CALLOSCORES¹'ACGT')/CALLOSCORES  © REMOVE SCORES THAT A
RE '*' OR '.'
LALLOSCORES„(LALLOSCORES¹'ACGT')/LALLOSCORES  © DITTO
CSYMPSCORES„(CSYMPSCORES¹'ACGT')/CSYMPSCORES
LSYMPSCORES„(LSYMPSCORES¹'ACGT')/LSYMPSCORES

ALLSCORES„CALLOSCORES,LALLOSCORES,CSYMPSCORES,LSYMPSCORES
ALLELEIND„'ACGT'¹(ALLSCORES)

—(1=+/ALLELEIND)/' FIXEDSYN„FIXEDSYN+1 ª …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)/¼4
ALLELE1„'ACGT'[IND]

FREQ1„(+/(CALLOSCORES=ALLELE1))÷(½CALLOSCORES)    © CALC ALLELE FREQ
UENCIES IN DIFFERENT SAMPLES
FREQ2„(+/(LALLOSCORES=ALLELE1))÷(½LALLOSCORES)
FREQ3„(+/(CSYMPSCORES=ALLELE1))÷(½CSYMPSCORES)
FREQ4„(+/(LSYMPSCORES=ALLELE1))÷(½LSYMPSCORES)

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†½‘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„(CALLOSCORES¹'ACGT')/CALLOSCORES  © REMOVE SCORES THAT A
RE '*' OR '.'
LALLOSCORES„(LALLOSCORES¹'ACGT')/LALLOSCORES  © DITTO
CSYMPSCORES„(CSYMPSCORES¹'ACGT')/CSYMPSCORES
LSYMPSCORES„(LSYMPSCORES¹'ACGT')/LSYMPSCORES

ALLSCORES„CALLOSCORES,LALLOSCORES,CSYMPSCORES,LSYMPSCORES
ALLELEIND„'ACGT'¹(ALLSCORES)
—(1=+/ALLELEIND)/' FIXEDNON„FIXEDNON+1 ª …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)/¼4
ALLELE1„'ACGT'[IND]

FREQ1„(+/(CALLOSCORES=ALLELE1))÷(½CALLOSCORES)    © CALC ALLELE FREQ
UENCIES IN DIFFERENT SAMPLES
FREQ2„(+/(LALLOSCORES=ALLELE1))÷(½LALLOSCORES)
FREQ3„(+/(CSYMPSCORES=ALLELE1))÷(½CSYMPSCORES)
FREQ4„(+/(LSYMPSCORES=ALLELE1))÷(½LSYMPSCORES)

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†½‘SYNSCORES)-(SYNCOUNT+FIXEDSYN)
NONLOWER„(1†½‘NONSCORES)-(NONCOUNT+FIXEDNON)
ŒTCLF
'M-K TABLE FOR CUTOFFS 1 AND 2 ',CUTOFF1,CUTOFF2
ŒTCLF
'           SYN     NONSYN'

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

ALPHA„1-(SYNFOCUS×NONLOWER)÷(NONFOCUS×SYNLOWER)
NUMBER„ALPHA×NONFOCUS
ŒTCLF
'ALPHA, NUMBER ',ALPHA,NUMBER
ŒTCLF
MATRIX„2 2½SYNFOCUS,NONFOCUS,SYNLOWER,NONLOWER
GTEST1 MATRIX
ŒTCLF
'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 3½0
DIM„1†½‘SCORES

TRIFNUCS„DIM½' '     ©  THIS WILL HOLD TRIFIDA NUCLEOTIDES; ORDER COR
RESPONDS TO ORDER IN ‘SCORES
TYPESNP„DIM½' '      ©  THIS WILL HOLD TYPE OF SNP (E.G. SYN, NONSYN,
 OTHER)

CURRCONTIG„0

I„0

©  READ IN FIRST PART OF FILE

'C:\TRIFLAC.TXT' ŒNTIE ¯1
BUFFER„ŒNREAD (¯1,82,3000000,STARBYTE)      ©  READ IN 3M CHARACTERS
ŒNUNTIE ¯1

IND11„(BUFFER=ŒAV[11])/¼½BUFFER             ©  MAKE AN INDEX OF WHERE
ŒAV[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)^(‘POSˆOLDEND)      © 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)/¼½IND3                      © SNP NUMBER FOR THOSE SNPS (I.
E. POSITION IN ‘SCORES)

…(0=½NUMS)/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„½NUMS
K„0
RETK:K„K+1
ŒTCLF
'*******************************************************************
***********************'

ŒTCLF
'PROCESSING SNP'

CURNUM„NUMS[K]
CURPOS„‘POS[CURNUM]

LACSEQ2„(OLDSTART½'-'),LACSEQ
TRIFSEQ2„(OLDSTART½'-'),TRIFSEQ


UP3: ŒTCLF
'LAC, TRIF, AND GENOME SEQS'

SEQ3„OLDSTART‡‘SEQUENCE
SIZE2„˜/(OLDSIZE,200)
LACSEQ[¼SIZE2]
TRIFSEQ[¼SIZE2]
```

```
SEQ3[¼SIZE2]

SC„,‘SCORES[CURNUM;;]
LACSNP„LACSEQ2[CURPOS]
TRIFSNP„TRIFSEQ2[CURPOS]
GENSNP„‘SEQUENCE[CURPOS]

ŒTCLF
'CONTIG ',OLDCONTIG,'  PROCESSING ',(1+(STARBYTE÷3000000)),'TH 3MB S
EGMENT'
BPPROC„(3000000-½BUFFER)
'      ',BPPROC,' BASES OF SEGMENT PROCESSED'
ŒTCLF
'SNP SCORES'
SC
ŒTCLF
'LAC, TRIF, AND GENOME NUCLEOTIDE'
LACSNP
TRIFSNP
GENSNP

IND21„(TRIFSEQ¬'-')
LACSEQ3„IND21/LACSEQ
TRIFSEQ3„IND21/TRIFSEQ

PCTEQ„(+/LACSEQ3=TRIFSEQ3)÷(½LACSEQ3)
'PCTEQ ',PCTEQ
–(PCTEQ<.5)/'TRIFNUCS[CURNUM]„''M'' ª ''M INSERTED'' ª …DOWN4'
       © INSERTS M INTO TRIFNUCS TO SIGNIFY UNRELIABLE ALIGNMENT

TEST3„LACSNP¹SC
'TEST3  ',TEST3
–(TEST3=0)/'TRIFNUCS[CURNUM]„''N'' ª ''N INSERTED''ª …DOWN4'
     ©  INSERTS N INTO TRIFNUCS TO SIGNIFY MISMATCH BETW ALLELES AND
 LACSNP

SEQ4„SEQ3[¼OLDSIZE]
PCTEQ2„(+/LACSEQ=SEQ4)÷(½LACSEQ)
'PCTEQ2 ',PCTEQ2
LINEINFO„(CURNUM,CURPOS,OLDCONTIG)

–(PCTEQ2<0.9)/'BADCONTIG„BADCONTIG,[1]LINEINFO ª ''BADCONTIG'' '
                                                    ©  INDICATES MISAL
IGNEMENT BETW LAC SEQ AND GENOME SCAFFOLD
TRIFNUCS[CURNUM]„TRIFSNP

ŒTCLF
'INSERTED NUCLEOTIDE: ',TRIFSNP

DOWN4:TYPESNP[CURNUM]„TYPE

TRSH„ŒDL 2
…(K<DIM2)/RETK

DOWN2:  © 'BUFFER BEFORE RETURN'
```

```
DOWN4:…(100000<½BUFFER)/DOWN5
STARBYTE„STARBYTE+3000000
'C:\TRIFLAC.TXT' ŒNTIE ¯1
BUFFER„BUFFER,ŒNREAD (¯1,82,3000000,STARBYTE)      ©  READ IN 3M MORE
 CHARACTERS
ŒNUNTIE ¯1

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

DOWN5:…UP2

ŒTCLF
'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=ŒAV[11])/¼½BUFFER)                      © POSITION OF F
IRST ŒAV[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=ŒAV[10])/¼½INFO                     ©  MAKE INDEX OF WHERE ŒAV
[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,ŒAV)[(LOWER,ŒAV)¼SEQUENCE]
END„¯1†SEQUENCE
–(END¹(ŒAV[10 11]))/'SEQUENCE„¯1‡SEQUENCE'

SOURCESIZE„ŒFI ¯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„ŒFI ¯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„ŒFI ¯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„ŒFI (TEMP¹'0123456789')/TEMP              ©  CONTIG NUMBER

TEST„+/(TEMP ŒSS 'lac')
–(TEST=1)/'SPECIES„''L'''

TEST„+/(TEMP ŒSS '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 ‘SEQUENCE

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

© BEFORE RUNNING THIS, MAKE SURE TO RUN 'CONVLACINDEX' TO CONVERT ‘L
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„‘LACINDEX[;1]
IND„(SCAFFN=SCAFFNUMS)/¼½SCAFFNUMS
TEMP1„,‘LACINDEX[IND;]
STARTBYTE„TEMP1[2]
ENDBYTE„TEMP1[3]


'C:\LACGENOME' ŒNTIE ¯1
 SCAFFOLD„ŒNREAD ¯1 82 (ENDBYTE-STARTBYTE) STARTBYTE
ŒNUNTIE ¯1
SCAFFNAME„24†SCAFFOLD
‘SEQUENCE„24‡SCAFFOLD
‘SEQUENCE„(‘SEQUENCE¬ŒTCLF)/‘SEQUENCE
************************************
```

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„½Y

MERGED„0½' '

I„0
RETI:I„I+1

X1„X[I]
Y1„Y[I]

–(X1¹'ACGT')/'MERGED„MERGED,X1 ª  …DOWN'
–(Y1¹'ACGT')/'MERGED„MERGED,Y1 ª …DOWN'
```

```
MERGED„MERGED,' '

DOWN:…(I<MAXI)/RETI

ŒTCLF
'TOTAL SNPS IDENTIFIED = ',+/MERGED¹'ACGT'
ŒTCLF

'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:

Co1: 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„½‘MERGED     © ‘MERGED IS SAME AS 'MERGED' PRODUCED BY PROGRAM '
MERGE'

NUMS„ŒFI  ,‘LACCODONS[;13+¼6]    © SNP NUMBERS FROM ‘LACCODONS

CSYNFREQS„LSYNFREQS„CNONFREQS„LNONFREQS„CRFREQS„LRFREQS„0½0    ©  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„(CSC¹'ACGT')/CSC         © GET RID OF '.'
LSC„,SC[‘LINDEX;]            © PICK OUT LAC SCORES
LSC„(LSC¹'ACGT')/LSC         © GET RID OF '.'
BOTH„CSC,LSC                 © COMBINE CORD AND LAC SCORES

TNUC„‘MERGED[I]             © PICK CORRESPONDING MERGED NUC
…(TNUC¹' NM')/DOWN           ©  IF NUC¹NM, SKIP
```

```
SNP„‘LACSNPS[I;10]          © READ SNP NUMBER FROM ‘LACSNPS
TEST„SNP¹NUMS               © TEST IF SNP NUMBER IN ‘LACCODONS
…(TEST=0)/DOWN              © IF NOT, SKIP

IND„(NUMS=SNP)/¼½NUMS       © POSITION OF SNP IN ‘LACCODONS
TYPE„‘LACCODONS[IND;12]     © GET TYPE (NON-SYN, SYN, ETC.

…(TYPE='O')/DOWN            © SKIP IF TYPE='O'

TEST„(TNUC¹BOTH)            © TEST IF MERGED ALLELE IN LAC OR CORD (AN
CESTRAL ALLELE)
…(TEST=0)/DOWN              © SKIP IF NOT


CFREQ„(+/CSC=TNUC)÷½CSC     © ANCESTRAL ALLELE FREQ IN CORD
LFREQ„(+/LSC=TNUC)÷½LSC     © 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 ª LNONFREQS„LNONFREQS,LFREQ'
–(TYPE='S')/'CSYNFREQS„CSYNFREQS,CFREQ ª LSYNFREQS„LSYNFREQS,LFREQ'
–(TYPE='R')/'CRFREQS„CRFREQS,CFREQ ª LRFREQS„LRFREQS,LFREQ'


DOWN:…(I<MAXI)/RETI

© BIN THE FREQUENCIES

NUMBINS„50    © CHANGE THIS TO WHATEVER
BINS„(¼NUMBINS)÷NUMBINS
BINS„BINS-BINS[1]

NBINS„+/CNONFREQS°.‰BINS
CNONTOT„+/[1]NBINS°.=(¼NUMBINS)

NBINS„+/LNONFREQS°.‰BINS
LNONTOT„+/[1]NBINS°.=(¼NUMBINS)

NBINS„+/CSYNFREQS°.‰BINS
CSYNTOT„+/[1]NBINS°.=(¼NUMBINS)

NBINS„+/LSYNFREQS°.‰BINS
LSYNTOT„+/[1]NBINS°.=(¼NUMBINS)

NBINS„+/CRFREQS°.‰BINS
CRTOT„+/[1]NBINS°.=(¼NUMBINS)

NBINS„+/LRFREQS°.‰BINS
LRTOT„+/[1]NBINS°.=(¼NUMBINS)

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„((¼NUMBINS)÷NUMBINS)-(1÷(2×NUMBINS))  © MIDPOINTS OF BINS

COUNT„+/BINS<.9     © NUMBER OF BINS WITH FREQ ˆ .9

MATRIX„(COUNT 1½MIDPOINTS),(COUNT 1½ALPHA1A),(COUNT 1½ALPHA1B),(COUN
T 1½ALPHA2A),(COUNT 1½ALPHA2B)

© CONVERT DATA TO MATRICES FOR MATHEMATICA

TMP„(COUNT 1½MIDPOINTS),(COUNT 1½ALPHA1A)
CONVMATH2 TMP
A1A„CONVDATA

TMP„(COUNT 1½MIDPOINTS),(COUNT 1½ALPHA1B)
CONVMATH2 TMP
A1B„CONVDATA

TMP„(COUNT 1½MIDPOINTS),(COUNT 1½ALPHA2A)
CONVMATH2 TMP
A2A„CONVDATA

TMP„(COUNT 1½MIDPOINTS),(COUNT 1½ALPHA2B)
CONVMATH2 TMP
A2B„CONVDATA

SKIP:

ŒTCLF
```

```
'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;

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

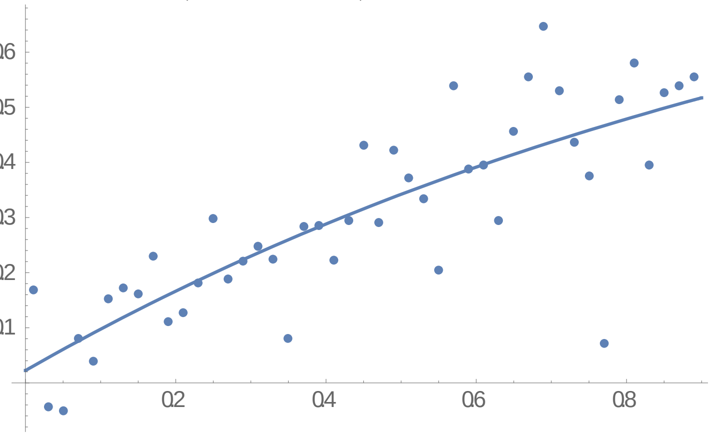
5. Estimating α(1) (asymptotic value of α) using MATHEMATICA. The data in vectors A1A, A1B, A2A and A2B are cut and pasted into the MATHEMATICA program (highlighted in yellow below), which produces estimates of the three regression parameters, and also plots the data and the fitted regression.

```
***********************************
(*  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},{0.13,0.1730674342},{0.15,0.1619181287},{0.17,0.2295005028},{0.19,0.1105263158},{0.21,0.1266399695},{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},{0.39,0.2853361728},{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},{0.65,0.455424275},{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[$\boxed{0.958508-0.936118e^{-0.835267y}}$]

{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 1½MIDPOINTS)

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='¯')/¼½MMAT1A
MMAT1A[IND]„'-'

'C:\MMAT1A' ŒNCREATE ¯1
MMAT1A ŒNAPPEND ¯1
ŒNUNTIE ¯1

MMAT1B„,•MAT1B
IND„(MMAT1B='¯')/¼½MMAT1B
MMAT1B[IND]„'-'

'C:\MMAT1B' ŒNCREATE ¯1
MMAT1B ŒNAPPEND ¯1
ŒNUNTIE ¯1

MMAT2A„,•MAT2A
IND„(MMAT2A='¯')/¼½MMAT2A
MMAT2A[IND]„'-'

'C:\MMAT2A' ŒNCREATE ¯1
MMAT2A ŒNAPPEND ¯1
ŒNUNTIE ¯1

MMAT2B„,•MAT2B
IND„(MMAT2B='¯')/¼½MMAT2B
MMAT2B[IND]„'-'

'C:\MMAT2B' ŒNCREATE ¯1
MMAT2B ŒNAPPEND ¯1
ŒNUNTIE ¯1

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

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

©  THIS PROGRAM BOOTSTRAPS THE ALPHA VALUES

DIM„½CNONFREQS
```

```
IND„?DIM½DIM
BCNONFREQS„CNONFREQS[IND]  © NOTE: CNONFREQS, ETC. FROM PREVIOUS RUN
 OF SEPMERGEDNUCS
BLNONFREQS„LNONFREQS[IND]

DIM„½CSYNFREQS
IND„?DIM½DIM
BCSYNFREQS„CSYNFREQS[IND]
BLSYNFREQS„LSYNFREQS[IND]

DIM„½CRFREQS
IND„?DIM½DIM
BCRFREQS„CRFREQS[IND]
BLRFREQS„LRFREQS[IND]

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

NBINS„+/BCNONFREQS°.>BINS
CNONTOT„+/[1]NBINS°.=(¼NUMBINS)

NBINS„+/BLNONFREQS°.>BINS
LNONTOT„+/[1]NBINS°.=(¼NUMBINS)

NBINS„+/BCSYNFREQS°.>BINS
CSYNTOT„+/[1]NBINS°.=(¼NUMBINS)

NBINS„+/BLSYNFREQS°.>BINS
LSYNTOT„+/[1]NBINS°.=(¼NUMBINS)

NBINS„+/BCRFREQS°.>BINS
CRTOT„+/[1]NBINS°.=(¼NUMBINS)

NBINS„+/BLRFREQS°.>BINS
LRTOT„+/[1]NBINS°.=(¼NUMBINS)

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 1½MIDPOINTS),(COUNT 1½ALPHA1A),(COUNT 1½ALPHA1B),(COUN
T 1½ALPHA2A),(COUNT 1½ALPHA2B)
© NOTE: COUNT IS FROM PROGRAM SEPMERGEDNUCS

© CONVERT DATA TO MATRICES FOR MATHEMATICA

TMP„(COUNT 1½MIDPOINTS),(COUNT 1½ALPHA1A)
CONVMATH TMP
A1A„CONVDATA

TMP„(COUNT 1½MIDPOINTS),(COUNT 1½ALPHA1B)
CONVMATH TMP
A1B„CONVDATA

TMP„(COUNT 1½MIDPOINTS),(COUNT 1½ALPHA2A)
CONVMATH TMP
A2A„CONVDATA

TMP„(COUNT 1½MIDPOINTS),(COUNT 1½ALPHA2B)
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†½‘SYNSCORES
ALD1„0 2½0
DIMSYMP„½‘CSYMPINDEX
DATA„0 4½0
MAXI„840

I„0
RETI:I„I+1            © LOOP FOR CONTIGS
ŒTCLF
'I = ',I

TIG„‘UNIQUETIGS[I]    © PICK OUT CONTIG NUMBER

SCINDEX„(‘SYNSNPS[;1]=TIG)/¼SNPNUM    ©  INDEX OF SNPS ON CONTIG

…(1‰½SCINDEX)/DOWN

SC„‘SYNSCORES[SCINDEX;;]     © PICK SCORES FOR EACH SNP ON CONTIG
SPOS„‘SYNSNPS[SCINDEX;3]     © PICK POSITIONS OF EACH SNP ON CONTIG

MAXSNP„½SCINDEX   © 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'¹UNSC1CS)
```

```
COL1„SC1CA[;1]       © CONVERT GENOTYPES TO X'S FOR CORD ALLOPATRIC
IND1„COL1¹'ACGT'
COL2„SC1CA[;2]
IND2„COL2¹'ACGT'
IND3„IND1^IND2
IND4„(IND3)/¼½IND3
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)÷(2×½XCA)   ©  ALLELE FREQUENCY IN CORDAT ALLOPATRIC

COL1„SC1LA[;1]     ©  CONVERT GENOTYPES TO X'S FOR LAC ALLOPATRIC
IND1„COL1¹'ACGT'
COL2„SC1LA[;2]
IND2„COL2¹'ACGT'
IND3„IND1^IND2
IND4„(IND3)/¼½IND3
SC1LA„SC1LA[IND4;]
Q„SC1LA=ALLELE1
XLA„+/Q            ©  EACH GENOTYPE CONVERTED TO SUM OF NUMBER OF ALL
ELES GIVEN BY 'ALLELE'
FREQ1LA„(+/XLA)÷(2×½XLA)   ©  ALLELE FREQUENCY IN LAC ALLOPATRIC


COL1„SC1CS[;1]     ©  CONVERT TENOTYPES TO X'S FOR CORD SYMPATRIC
IND1„COL1¹'ACGT'
COL2„SC1CS[;2]
IND2„COL2¹'ACGT'
IND3„IND1^IND2
IND4„(IND3)/¼½IND3
SC1CS„SC1CS[IND4;]
Q„SC1CS=ALLELE1
XCS„+/Q            ©  EACH GENOTYPE CONVERTED TO SUM OF NUMBER OF ALL
ELES GIVEN BY 'ALLELE'
FREQ1CS„(+/XCS)÷(2×½XCS)   ©  ALLELE FREQUENCY IN CORD SYMPAtRIC

ALL1„,(SC1CA,[1]SC1LA),[1]SC1CS
TEST1„+/'ACGT'¹ALL1
…(TEST1ˆ1)/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„COL1¹'ACGT'
COL2„SC2CA[;2]
IND2„COL2¹'ACGT'
IND3„IND1^IND2
IND4„(IND3)/¼½IND3
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)÷(2×½XCA2)   ©  ALLELE FREQUENCY IN CORDAT ALLOPATRI
C

COL1„SC2LA[;1]     ©  CONVERT TENOTYPES TO X'S FOR LAC ALLOPATRIC
IND1„COL1¹'ACGT'
COL2„SC2LA[;2]
IND2„COL2¹'ACGT'
IND3„IND1^IND2
IND4„(IND3)/¼½IND3
SC2LA„SC2LA[IND4;]
Q„SC2LA=ALLELE2
XLA2„+/Q           ©  EACH GENOTYPE CONVERTED TO SUM OF NUMBER OF AL
LELES GIVEN BY 'ALLELE'
FREQ2LA„(+/XLA2)÷(2×½XLA2)   ©  ALLELE FREQUENCY IN LAC ALLOPATRIC

COL1„SC2CS[;1]     ©  CONVERT TENOTYPES TO X'S FOR CORD SYMPATRIC
IND1„COL1¹'ACGT'
COL2„SC2CS[;2]
IND2„COL2¹'ACGT'
IND3„IND1^IND2
IND4„(IND3)/¼½IND3
SC2CS„SC2CS[IND4;]
Q„SC2CS=ALLELE2
XCS2„+/Q           ©  EACH GENOTYPE CONVERTED TO SUM OF NUMBER OF AL
LELES GIVEN BY 'ALLELE'
FREQ2CS„(+/XCS2)÷(2×½XCS2)   ©  ALLELE FREQUENCY IN CORD SYMPARIC

ALL1„,(SC2CA,[1]SC2LA),[1]SC2CS
TEST2„+/'ACGT'¹ALL1
…(TEST2ˆ1)/DOWN3        © SKIP SNP IF NOT POLYMORPHIC
TEST„+/('ACGT'¹UNSC2CS)

MEANX2„2×FREQ2CS

© CALCULATE COVAR(SNP1,SNP2)

SUM„0
COUNT„0
L„0
```

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

TEST„('*'¹(G1,G2))Ÿ('.'¹(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)/'ŒTCLF ª ''COV GREATER THAN 1.2'' ª …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†½BINS
ALTMEANW„0½0

ALDBINS„0 5½0
X„‘CALDDATAB2   © CHANGE FILE ACCORDINGLY
COUNTS„0½0

I„0
RETI:I„I+1
```

```
TEST„(I÷100)=(—(I÷100))
—(TEST=1)/'I'

COUNT„IND„+/(X[;2]ˆ(BINS[I;2]))
COUNTS„COUNTS,IND
PART„X[¼IND;]

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)/¼½REDPART
ALTM„(+/REDPART[IND])÷½IND
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

‘BINS„1 2½(1,1000)
I„0
RETI:I„I+1
START„‘BINS[I;2]+1
END„—(START-1)+1000×X*I
‘BINS„‘BINS,[1](START,END)
…(END<4500000)/RETI
***********************************
```

VII. Calculation of hybrid indices was done using script HYBINDEX. It produces a matrix of hybrid indices that can be imported into SAS for analysis.

```
**********************************
HYBINDEX
©  THIS PROGRAM CALCULATES THE DISTRIBUTION OF HYBRID INDICES FOR SY
MPATRIC SAMPLES.


ROWS1„½‘CSYMPINDEX       © ‘SYMPINDEX IS INDEX OF CORDAT SYMPATRIC SA
MPLES
ROWS2„½‘LSYMPINDEX       © ‘LSYMPINDEX IS INDEX OF LAC SYMPATRIC SAMP
LES

CINDEX„(ROWS1, 0)½0
LINDEX„(ROWS2, 0)½0

DIM„1†½‘SCORES       ©  CHANGE FOR ‘SCORES, ‘SYNSCORES, ‘NONSCORES,
OR ‘RSCORES

I„0
RETI:I„I+1
TEST„(I×1000)-(–(I×1000))
–(TEST=1)/'I'

SC„‘SCORES[I;;]       ©  CHANGE FOR ‘SCORES, ‘SYNSCORES, ‘NONSCORES,
OR ‘RSCORES

CSC„SC[‘CALL03;]
LSC„SC[‘LALL03;]
CSC2„((,CSC)¹'ACGT')/,CSC
LSC2„((,LSC)¹'ACGT')/,LSC

…(0=½CSC2)/DOWN
ALLELE„CSC2[1]

CFREQ„(+/CSC2=ALLELE)÷(½CSC2)
LFREQ„(+/LSC2=ALLELE)÷(½LSC2)
DIFF„|(CFREQ-LFREQ)

…(DIFF¬1)/DOWN

CSYMP„SC[‘CSYMPINDEX;]
LSYMP„SC[‘LSYMPINDEX;]

C1„CSYMP=ALLELE
C2„9×CSYMP¹'.*'
C„C1+C2
L1„LSYMP=ALLELE
L2„9×LSYMP¹'.*'
L„L1+L2
LINDEX„LINDEX,L
```

```
DOWN:…(I<DIM)/RETI

SIZE„1‡½CINDEX
NINES„+/(CINDEX=9)
SUM„+/CINDEX
SUM2„SUM-(9×NINES)
NUMS„SIZE-NINES
CHINDEX„1-(SUM2÷NUMS)

SIZE„1‡½LINDEX
NINES„+/(LINDEX=9)
SUM„+/LINDEX
SUM2„SUM-(9×NINES)
NUMS„SIZE-NINES
LHINDEX„SUM2÷NUMS

ŒTCLF
'HYBRID INDICES FOR SYMPATRIC I. CORDATOTRILOBA'
CHINDEX
ŒTCLF
'HYBRID INDICES FOR SYMPATRIC I. LACUNOSA'
LHINDEX         © 1 - LHINDEX SHOWN TO MAKE COMPARABLE TO CHINDEX

©  MAKE DATA FOR SAS

TMP„(ROWS1,1)½CHINDEX
TMP2„(ROWS2,1)½(LHINDEX)
SASDATA„(((ROWS1,1)½1),TMP),[1](((ROWS2,1)½2),TMP2)

ŒTCLF
'PROGRAM HYBINDEX FINISHED.  DATA IN VARIABLES CHINDEX, LHINDEX, AND
 SASDATA.'
******************************
```

VIII. Simulating gene flow. The script SWAPFIT2 swaps different proportions of 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 SS across 20 replicate simulations for each proportion. The proportion with the lowest mean SS is taken as the best estimate of the true proportion. The script also The script SWAPFIT2 calls three other scripts: SWAPTIG2, FREQCHANGEP, FREQCHANGEANAL, and CALCSS8, which are also listed below.

```
**********************************
SWAPFIT2
© THIS PROGRAM EVALUATES THE FIT OF A MODEL WITH A GIVEN PROPORTOIN
OF SWAPS OF I. LAC
©   SYMPATRIC LOCI FOR I. CORD

COUNTSSTACK„COUNTS2STACK„0 21 11½0      © COUNTS AND COUNTS2 FOR ALL R
EPLICATES
MEANSTACK„MEANSTACK2„0 21 11½0
ALLSS„0½0                               ©  SUM OF SQUARES VALUES FOR A
LL REPLICATES
ALLSS8„0½0
PROPNUMS„.3,.4,.45, (.45+ (¼20)÷100),.7,.8,.9 1             ©  DEFIN
E VECTOR OF PROPS OF LOCI SWAPPED
DIM2„½PROPNUMS
MEANSS„0½0              ©  MEAN SS FOR A GIVEN VALUE OF PROP
VARSS„0½0               ©  VAR OF SS FOR A GIVEN VALUE OF PROP
MEANSS8„0½0

IJ„0                                    © LOOP FOR DIFFERENT PROPS
RETIJ:IJ„IJ+1
ŒTCLF

PROP„PROPNUMS[IJ]      © SETS PROPORTION OF LOCI TO SWAP

SMALLSTACK„SMALL2STACK„0 21 11½0      © COUNTS AND COUNTS2 FOR A GIVEN
 VALUE OF PROP

REPS„20                 ©  NUMBER OF REPS PER VALUE OF PROP
IK„0                    ©  REP LOOP
RETIK:IK„IK+1
ŒTCLF
'RUNNING PROP = ',PROP,'  REP = ',IK

SWAPTIG2                   © SWAPS LOCI. USE SWAPTIG FOR KNOWN ALLO, SW
APTIG2 FOR CLOSE ALLO
FREQCHANGEP             © CALCULATES FREQCHANGE FOR EACH LOCUS
FREQCHANGEANAL          © ACCUMULATES FREQ CHANGES INTO BINS
COUNTSSTACK„COUNTSSTACK,[1]COUNTS       © SAVES COUNTS
COUNTS2STACK„COUNTS2STACK,[1]COUNTS2    © SAVE COUNTS2

SMALLSTACK„SMALLSTACK,[1]COUNTS         © SAVE COUNTS FOR DIFF REPS FO
R GIVEN VALUE OF PROP
```

```
SMALL2STACK„SMALL2STACK,[1]COUNTS2

SS1„+/+/((COUNTS-‘COUNTSOBS)*2)          ©  CALCULATE SUM OF SQUARES
 FOR EACH REP
ALLSS„ALLSS,SS1                          ©  ADD SS TO ALLSS

CALCSS8 COUNTS
ALLSS8„ALLSS8,SS8

ŒTCLF
'SUM OF SQUARES FOR PROP = ',PROP,' REP = ',IK,',IS ',SS1
ŒTCLF
'SS8 FOR PROP = ',PROP,' REP = ',IK,',IS ',SS8
…(IK<REPS)/RETIK


©  LOOP FOR CALCULATING MEAN AND VAR OF SS FOR GIVEN VALUE OF PROP

REPSS„0½0      © VECTOR OF REP SS'S
REPSS8„0½0

KI„0
RETKI:KI„KI+1

REPCOUNTS„SMALLSTACK[KI;;]
SSREP„+/+/(REPCOUNTS-‘COUNTSOBS)*2      © CALCULATE SS FOR REP
REPSS„REPSS,SSREP                       ©  APPEND REP SS TO REPSS

CALCSS8 REPCOUNTS                       © CALCULATE SS8 FOR REP
REPSS8„REPSS8,SS8                       © APPEND REP SS8 TO REPSS8

…(KI<REPS)/RETKI

MEAN„(+/REPSS)÷REPS                    © CALCULATE MEAN OVER REP SS'S
MEANSS„MEANSS,[1]MEAN
DIFFS„MEAN-REPSS                       © CALCULATE VARIANCE OVER REP
SS'S
VAR„(+/DIFFS*2)÷(REPS-1)
VARSS„VARSS,VAR
MEAN8„(+/REPSS8)÷REPS
MEANSS8„MEANSS8,MEAN8


MEANCOUNTS„(+/[1]SMALLSTACK)÷REPS        © MEAN OF COUNTS ACROSS REP
S FOR A GIVEN VALUE OF PROP
MEANCOUNTS2„(+/[1]SMALL2STACK)÷REPS
MEANSTACK„MEANSTACK,[1]MEANCOUNTS       © ADD MEAN COUNTS TO MEANSTA
CK
MEANSTACK2„MEANSTACK2,[1]MEANCOUNTS2

ŒTCLF
'MEAN SS FOR PROP = ',PROP,' IS ',MEANSS
'STANDARD ERROR = ',(VAR*.5)
'MEAN SS8 FOR PROP= ',PROP,' IS ',MEANSS8
```

```
…(IJ<DIM2)/RETIJ

ŒTC
'PROGRAM SWAPFIT2 FINISHED. DATA IN VARIABLES COUNTSSTACK AND COUNTS
2STACK, ALLSS, AND MEANSS AND MEANSS8.'
ŒTCLF
'SUMS OF SQUARES'
TEMP„((DIM2,1)½PROPNUMS),((DIM2,1)½MEANSS),((DIM2,1)½VARSS)

ŒTCLF
'PROP        MEANSS      VARSS'
ŒTCLF
12 2•(TEMP)


10 0•4+¼5
10 0•MEANSS
ŒTCLF
'MEAN SS FOR ROWS 1 - 9.'

******************************
SWAPTIG2
© THIS FUNCTION TAKES A RANDOM SET OF CONTIG PARTS FROM CLOSE ALLOPA
TRIC I. LACUNOSA AND SUBITUTES THEM INTO
©    I. CORDATOTRILOBA
©  THIS CREATES A 'PSEUDO' SYMPATRIC POPULATION OF I. CORD THAT WILL
 THEN BE USED TO CALCULATE
©    PI AND DXY BETWEEN I. LAC CLOSE ALLOPATRIC AND I.CORD PSEUDO SY
MPATRIC POPULATION.

PSEUDOSCORES„‘SCORES
DIM1„1†½‘SCORES
BLOCKDIM„1†½‘BLOCKS

TOTLOCI„˜PROP×DIM1    © TOTAL NUMBER OF LOCI TO SWAP

IND1„BLOCKDIM?BLOCKDIM    © RANDOM ORDER OF BLOCKS TO PICK

SWAPNUMS„0½0    ©  VARIABLE TO HOLD LOCI NUMBERS TO BE SWAPPED

I„0
RETI:I„I+1
BLOCKNUM„IND1[I]           © PICK RANDOM BLOCK NUMBER
BLOCK„‘BLOCKS[BLOCKNUM;]   © CHOOSE ENDPOINTS OF BLOCK
LOCNUMS„(BLOCK[1]-1)+¼(1+BLOCK[2]-BLOCK[1])    © CALCULATE LOCI NUMBE
RS

DIFF„TOTLOCI-½SWAPNUMS    © CALCULATE MAXIMUM NUMBER OF LOCI TO ADD
NEWNUMS„DIFF†LOCNUMS      © PICK NEW LOCI NUMBERS
TEMP„(NEWNUMS>0)/¼½NEWNUMS    ©  GET RID OF ZEROS
NEWNUMS„NEWNUMS[TEMP]
SWAPNUMS„SWAPNUMS,NEWNUMS    ©ADD LOCUS NUMBERS TO SWAPNUMS

…(TOTLOCI>½SWAPNUMS)/RETI
```

```
SWAPNUMS„SWAPNUMS[“SWAPNUMS]    © ORDER LOCI TO BE SWAPPED

K„0                        © LOOP TO SUBSTITUTE LOCI
RETK:K„K+1

© REPLACE LSYMP SCORES WITH LALLO3 SCORES
LSYMP„‘SCORES[K;‘LALLOCLOSE;]      ©  ALTERNATELY ‘LALLO3
IND2„(¼9),4?9
PSEUDOSCORES[K;‘LSYMPINDEX;]„LSYMP[IND2;]

© REPLACE CSYMP SCORES WITH CALLO3 SCORES
CSYMP„‘SCORES[K;‘CALLOCLOSE;]     ©  ALTERNATELY ‘CALLO3
IND3„(¼9),2?9
PSEUDOSCORES[K;‘CSYMPINDEX;]„CSYMP[IND3;]

© TEST FOR WHETHER TO SWAP LALLO3 INTO CSYMP
TEST„K¹SWAPNUMS
…(TEST=0)/DOWN1

© REPLACE CSYMP SCORES WITH LALLO3 SCORES
LACTIGS„PSEUDOSCORES[K;‘LSYMPINDEX;]
IND2„11?13
CTIGS„LACTIGS[IND2;]

PSEUDOSCORES[K;‘CSYMPINDEX;]„CTIGS

DOWN1:…(K<DIM1)/RETK
*********************************
FREQCHANGEP
© THIS PROGRAM CALCULATES THE CHANGE IN FREQUENCY DIFFERENCE BETWEEN
 SYM LAC AND CORD AND ALLOPATRIC LAC AND CORDAT

©  RUN FRQCHANGEANAL AND FREQCHANGECONV AFTER THIS TO CONVERT DATA T
O FORMAT FOR MATHEMATICA

STATUS„‘LACCODONS[;12]     © COLUMN OF N'S, S'S, O'S
SNPNUMS„‘LACCODONS[;12+¼7]    © SNP NUMBERS FROM ‘LACCODONS IN TEXT
FORMAT
BLANKS„((½STATUS)½' ')        ©   ADD ONE COLUMN OF SPACES
SNPNUMS„,(SNPNUMS,BLANKS)
SNPNUMS2„ŒFI SNPNUMS          © CHANGE SNP NUMBERS TO NUMERIC

DIM„1†½PSEUDOSCORES             ©  USE ALL SNPS

‘FREQCHANGEDATA„0 8½0     ©  INITIALIZE VARIABLE TO HOLD DATA.  EACH
 LINE IS A VECTOR WITH ELEMENTS
                          ©      SCAFFOLD, SCAFFOLD POSITION, CALLOP
 FREQ, LALLOP FREQ, CSYMP FREQ, LSYMP FREQ,
                          ©      ALLOPATRID FREQ DIFF, SYMP FREQ DIF
F,
‘EXTREME„ 0 7½0           ©  VARIABLE TO HOLD INFO ON SYMP DIFFS > S
OME THRESHOLD
```

```
I„0
RETI:I„I+1
TEST„(I÷1000)=(–(I÷1000))
–(TEST)/'I'
© SCAFF„‘LACSNPS[I;1]          © GET SCAFF NUMBER OF SNP
© POS„‘LACSNPS[I;3]            © GET SCAFFOLD POSITION OF SNP
© CDS„‘LACSNPS[I;2 4]
© SNPNUM„‘LACSNPS[I;10]         © GET CDS BOUNDARIES FOR SNP
SCORES„PSEUDOSCORES[I;;]          ©  PICK SNP
CASCORES„SCORES[‘CALLO3;]     © DIVIDE SCORES INTO CALLO, LALLO, CSYM
P AND LSYMP
LASCORES„SCORES[‘LALLO3;]     © USE EITHER ‘CALLO3 OR ‘CALLOCLOSE, ‘L
ALLO3 OR ‘LALLOCLOSE
CSSCORES„SCORES[‘CSYMPINDEX;]
LSSCORES„SCORES[‘LSYMPINDEX;]

IND„(CASCORES[;1]¬'.')/¼1†½CASCORES    ©  GET RID OF MISSING DATA
CASCORES„CASCORES[IND;]

IND„(LASCORES[;1]¬'.')/¼1†½LASCORES
LASCORES„LASCORES[IND;]

IND„(CSSCORES[;1]¬'.')/¼1†½CSSCORES
CSSCORES„CSSCORES[IND;]

IND„(LSSCORES[;1]¬'.')/¼1†½LSSCORES
LSSCORES„LSSCORES[IND;]

ALLELE„CASCORES[1;1]                  ©  ARBITRARILY CHOOSE ALLELE

CAFREQ„(+/(,CASCORES)=ALLELE)÷(2×1†½CASCORES)    ©  CALCULATE ALLELE
FREQUENCY IN EACH SET OF SAMPLES
LAFREQ„(+/(,LASCORES)=ALLELE)÷(2×1†½LASCORES)
CSFREQ„(+/(,CSSCORES)=ALLELE)÷(2×1†½CSSCORES)
LSFREQ„(+/(,LSSCORES)=ALLELE)÷(2×1†½LSSCORES)

DIFFA„CAFREQ-LAFREQ                                ©  DIFFERENCE IN AL
LOPATRIC FREQUENCIES
DIFFS„CSFREQ-LSFREQ                                ©  DIFFERENCE IN SY
MPATRIC FREQUENCIES

–(DIFFA<0)/'DIFFA„¯1×DIFFA ª DIFFS„¯1×DIFFS'      ©  IF ALLOP DIFFERE
NCE IS NEGATIVE, MULTIPLY BOTH DIFFERENCES BY ¯1

‘FREQCHANGEDATA„‘FREQCHANGEDATA,[1](SCAFF,9,CAFREQ,LAFREQ,CSFREQ,LSF
REQ,DIFFA,DIFFS)  © APPEND DATA

IND„(SNPNUMS2=SNPNUM)/¼½SNPNUMS2
TEMP„STATUS[IND]
–(0=½TEMP)/'…DOWN'
–(TEMP='R')/'SYNNONSYN„2'
–(TEMP='N')/'SYNNONSYN„1'
–(TEMP='S')/'SYNNONSYN„0'
```

```
–((DIFFA‰0.8)^(DIFFS‰0.5))/''‘EXTREME„‘EXTREME,[1](SNPNUM, SYNNONSYN,
 SCAFF, 9, CDS,DIFFS)'
DOWN:…(I<DIM)/RETI
ŒTCLF
'PROGRAM FREQCHANGEP COMPLETED.  DATA IN VARIABLE ‘FREQCHANGEDATA.'
ŒTCLF
'RUN ''FREQCHANGEANAL'' AND THEN ''FREQCHANGECONV'' TO CONVERT DATA
FOR MATHEMATICA INPUT.'

******************************
FREQCHANGEANAL
© THIS PROGRAM TAKES OUTPUT FROM FREQCHANGE AND PUTS COUNTS INTO BIV
ARIATE BINS

DIFF1„‘FREQCHANGEDATA[;7]    © VECTOR OF ALLOPATRIC FREQ DIFFS
DIFF2„‘FREQCHANGEDATA[;8]    © VECTOR OF SYMPATRIC FREQ DIFFS
BINSY„¯1.15+(.1×¼21)         © MAKE BINS BOUNDARIES FOR SYMPATRIC DIF
FERENCES
DIM„1†½‘FREQCHANGEDATA               © MAX FOR LOOP I
COUNTS„21 11½0               © INITIALIZE VARIABLE THAT WILL HOLD COU
NTS OF PARTICULAR
                             ©   ALLOPATRIC AND SYMPATRIC FREQ DIFFER
ENCES
BINSX„¯.15+0.1×¼11           © MAKE BIN BOUNDARIES FOR ALLOPATRIC DIF
FS

I„0                          ©  LOOP TO MAKE SNP COUNTS FOR COMBINATI
ONS OF ALLOP AND SYMP FREQ DIFFS
RETI:I„I+1
LINE„‘FREQCHANGEDATA[I;]     ©  PICK LINE FROM ‘FREQCHANGEDATA PRODUC
ED BY PROGRAM FREQCHANGE
                             ©      CORRESPONDS TO A PARTICULAR SNP
DIFF1„LINE[7]                ©  ALLOPATRIC FREQ DIFF FOR SNP
DIFF2„LINE[8]                ©  SYMPATRIC FREQ DIFF
X„(+/DIFF1‰BINSX)            ©  DETERMINE WHICH ALLOP DIFF BIN
Y„(+/DIFF2‰BINSY)            ©  DETERMINE SYMP DIFF BIN
Y„22-Y                       ©  REVERSE ORDER OF SYMP DIFF BINS
COUNTS[Y;X]„COUNTS[Y;X]+1    ©  ADD 1 TO APPROPRIATE CELL OF COUNTS
…(I<DIM)/RETI

COUNTS2„COUNTS               ©  BEGIN CHANGING COUNTS TO COLUMN PERCE
NTAGES (PCTGS OF SYMP DIFF FOR A GIVEN ALLOP DIFF BIN)
J„0
RETJ:J„J+1

COL„COUNTS2[;J]              © SYMP DIFF COUNTS FOR A ALLOP DIFF BIN
J
COL„100×COL÷(+/COL)          © CONVERT TO PERCENTAGE OF COLUMN COUNTS
COUNTS2[;J]„COL
…(J<11)/RETJ

SASCOUNTS„0 3½0              ©  THE FOLLOWING STATEMENTS TAKE DATA FR
OM COUNTS AND COUNTS2 AND PUT THEM
                             ©    IN FORMAT FOR SAS
```

```
                            ©  THIS PRODUCES A LINE FOR EACH ALLOP D
IFF - SYMP DIFF CELL THAT CONTAINS
                            ©     ALLOP DIFF BIN, SYMP DIFF BIN, (CO
UNTS OR COLUMN PCTGS)
SASPCTGS„0 3½0
K„0
RETK:K„K+1

L„0
RETL:L„L+1

LINE„BINSX[K],BINSY[L],COUNTS[L;K]
SASCOUNTS„SASCOUNTS,[1]LINE
LINE„BINSX[K],BINSY[L],COUNTS2[22-L;K]
SASPCTGS„SASPCTGS,[1]LINE

…(L<21)/RETL
…(K<11)/RETK


ŒTCLF
'PROGRAM FREQCHANGEANAL FINISHED.  DATA IN MATRIX COUNTS AND COUNTS2
.'
'DATA FOR SAS IN VARIABLES SASCOUNTS AND SASPCTGS.'
******************************
CALCSS8  Y

X„‘COUNTSOBS[;¼9]
Z„Y[;¼9]
DIFFS„X-Z
SS8„+/+/DIFFS*2
******************************
```