Appendix I: R code for quantifying multi-core lake-level reconstructions

We have implemented our decision-tree approach (Fig. 2) in R (R Core Development Team, 2009), and provide the code here, which was run on version 3.0.1. The analysis for Lake of the Woods was based on a single data file (LOW_Cores.csv), and a file of facies thresholds equivalent to Table 1 (FaciesThresholds.csv). All files are available from the NOAA Paleoclimate Program Data Center.

For other sites, similar input files could be generated, and the code modified to account for the total number of cores ("Num.cores"), the number of facies threshold iterations to apply ("total.iterations"; consistent with the input file), the modern water depth of the sediment limit ("SedLim.modern"), the time span of the reconstruction in years ("Length"), and the interpolation interval in years ("Interp.interval"). The data files must include the elevation (combined water and sediment depth), median age, and percent sand (or other sedimentary characteristic) data for each 1-cm interval in each core; these data fields require the names "Depth", "Age", and "Sand" respectively. Samples in the file must be labeled by core using a depth rank (in a field labeled "Core.DepthRank") from 1 (deepest) to n (shallowest). Note that sand content data could be replaced by other equivalent datasets, such as was done with LOI at Little Windy Hill Pond, but the code should be modified accordingly (e.g., if LOI data were used, the thresholds should be changed from greater-than to lesser-than values).

The output of the code is a comma-deliminated (.csv) text file, which provides the mean and bounding limits of the ensemble of the reconstructions. The lake elevations are represented as differences from the modern level; positive values represent lower-than-modern elevations in centimeters (or another unit, such as meters, if used in the input file). The ensemble mean elevation (labeled as "LakeElev.Anom") is produced for each time step (excluding the first and

last step because of a 3-point moving average applied to the reconstruction), as are the means of the upper and lower bounds on the lake-elevation reconstructions, which represent the total uncertainty associated with both the core spacing (determining the facies elevations) and the facies thresholds (accurately interpreting the paleo-environments).

The code involves sixteen steps, which are separated by explanatory comments:

```
#1. Read in core data and facies thresholds
Core = read.csv("LOW_Cores.csv",header=T)
Thresh = read.csv("FaciesThresholds.csv",header=T)

#2. Provide inputs
Num.cores=2
total.iterations=16
SedLim.modern=100
Length=18000
Interp.interval=50

#3. Create interpolation intervals
Int50=((1:(Length/Interp.interval))*Interp.interval)-
Interp.interval

#4. Interpolate core data
Data.i=matrix(NA,length(Int50),Num.cores)
Elev.i=matrix(NA,length(Int50),Num.cores)

i=1

while(i<Num.cores+1){

Core.i=approx(Core$Age[Core$Core.DepthRank==i],Core$Sand[Core$Co
re.DepthRank==i],method="linear",Int50)
Core.depth.i=approx(Core$Age[Core$Core.DepthRank==i],Core$Depth[
Core$Core.DepthRank==i],method="linear",Int50)

Data.i[,i]=Core.i$y
Elev.i[,i]=Core.depth.i$y

i=i+1

}

#5. Re-scale elevation relative to modern sediment limit
Elev.i=Elev.i-SedLim.modern
```

```
#6. Create matrix for storing iterative output
rows=length(Int50)-2

Recon.Iterations=matrix(NA,rows,total.iterations)
UpperBounds=matrix(NA,rows,total.iterations)
LowerBounds=matrix(NA,rows,total.iterations)

#7. Initiate loop to run all threshold iterations
iteration=1

while(iteration<total.iterations+1){

Lit.Threshold=Thresh[iteration,1]
Sublit.Threshold=Thresh[iteration,2]

#8. Identify facies
Core.lit = cbind(Data.i> Lit.Threshold)
Core.sublit = cbind(Data.i> Sublit.Threshold)
Core.profundal = cbind(Data.i<= Sublit.Threshold)

#9. Estimate littoral elevations
All.lit.max=apply(Core.lit*Elev.i,1,max,na.rm=TRUE)

#10. Estimate sublittoral elevation
All.sublit.max=apply(Core.sublit*Elev.i,1,max,na.rm=TRUE)

#11. Account for intervals with only littoral sediment in all
cores by setting a maximum bound on low water.

Only.lit=(All.lit.max==Elev.i[,1])*(All.sublit.max==All.lit.max)
All.sublit.max[Only.lit>0]=max(Elev.i,na.rm=T)

#12. Account for intervals with littoral but no sub-littoral
sediment by using the next deepest core below the littoral
sediments as a maximum limit on the sub-littoral facies.

NextDeepestCore=Core.profundal*Elev.i
NextDeepestCore[rowMeans(NextDeepestCore)==0]=max(Elev.i[rowMean
s(NextDeepestCore)==0])
NextDeepestCore[NextDeepestCore==0]=NA

DeepestCoreElev=apply(Elev.i,1,max,na.rm=T)
NextDeepestCore[is.na(NextDeepestCore)]=DeepestCoreElev[is.na(Ne
xtDeepestCore)]
NextDeepestCore=apply(NextDeepestCore,1,min,na.rm=T)
```

```
All.sublit.max[All.sublit.max==All.lit.max]=NextDeepestCore[All.
sublit.max==All.lit.max]
All.sublit.max[is.na(All.sublit.max)]=DeepestCoreElev[is.na(All.
sublit.max)]

#13. Account for intervals with no littoral or sub-littoral
sediment in all cores by using the elevation of the shallowest
core as a minimum limiting elevation for sub-littoral sediments.

ShallowestCoreElev=apply(Elev.i,1,min,na.rm=T)
All.sublit.max=All.sublit.max+((All.sublit.max==0)*ShallowestCor
eElev)

#14. Estimate sediment limit depths
All.sublit.smooth=(All.sublit.max[1:(length(Int50)-
2)]+All.sublit.max[2:(length(Int50)-
1)]+All.sublit.max[3:length(Int50)])/3
All.lit.smooth=(All.lit.max[1:(length(Int50)-
2)]+All.lit.max[2:(length(Int50)-
1)]+All.lit.max[3:length(Int50)])/3

SedLim = (All.sublit.smooth+All.lit.smooth)/2

#15. Add results to matrix of iterations and calculate ensemble
mean.
Recon.Iterations[,iteration]=SedLim
UpperBounds[,iteration]=All.lit.smooth
LowerBounds[,iteration]=All.sublit.smooth

iteration=iteration+1

}

SedLim = rowMeans(Recon.Iterations)
All.sublit.smooth=rowMeans(LowerBounds)
All.lit.smooth=rowMeans(UpperBounds)

SedLim.smooth=cbind(Int50[2:(length(Int50)-
1)],SedLim,All.lit.smooth, All.sublit.smooth)
colnames(SedLim.smooth)=c("Age","LakeElev.Anom","UpperBound","Lo
werBound")
SedLim.smooth=data.frame(SedLim.smooth)

#16. Write output file
write.csv(SedLim.smooth, "LLrecon.csv", row.names = FALSE)
```