Appendix

Script of SemGrid (1.3.0) for the application of the land evaluation procedure and the creation of the land suitability maps for medicinal herbs (`HerbLand.cmf`). This script accepts input layers in the ArcGis ascii grid format. To change the input type, substitute in the text the word "ArcGis" with "Geomedia", "Surfer" of other formats.

```
' === PROCEDURE FOR LAND EVALUATION FOR SUITABILITY FOR MEDICINAL PLANTS ======
'                       (HerbLand.cmf)
' INPUT:   Alt.txt          Altitude                        (m asl)   Ascii grid
'          Slope.txt        Slope                            (pc)     Ascii grid
'          Aspect.txt       ASpect                         (10-360°)  Ascii grid
'          Distrade.txt     Minimum distance cell-roads     (m)       Ascii grid
'          Disliv.txt       height difference cell-roads    (m)       Ascii grid
'          Usosuolo.txt     Use of the soil                           Ascii grid
'          ParTab.csv o .dct     Table of species parameters.  .CSV/.DCT
'          WeightTab.csv o .dct  table of species parameters.  .CSV/.DCT

' OUTPUT:  Va.txt     Ascii grid ArcGis   Agronomic suitability
'          Vi.txt     Ascii grid ArcGis   Quality suitability
'          Vt.txt     Ascii grid ArcGis   Environmental suitability

' ====================================================
'                   USE OF THIS SCRIPT:
'====================================================
' - Run SemGrid
' - In the command dialog, issue the following command:
'   herbland <species_table> <species_name> <weight_table> <weight_set_name>
' %1% = name of the table with species parameters with .csv extension
' %2% = name of the species in the table to be processed
' %3% = name of the table with different weight sets (.csv)
' %4% = name of the weight set to be used

' ========================================================
'       LOADS PARAMETERS FOR THE INDICATED SPECIES
' ========================================================
use %1%
scalar Irow=findrow(specie,"%2%")

' ------- Altitude -------------
' - environmental suitability
scalar PaltOptAD1=altOptAD1[Irow]
scalar PaltOptAD2=altOptAD2[Irow]
scalar PaltSDAD1=altSDAD1[Irow]
scalar PaltADAD2=altADAD2[Irow]

' yield
scalar PaltOptRE1=altOptRE1[Irow]
scalar PaltOptRE2=altOptRE2[Irow]
scalar PaltSDRE1=altSDRE1[Irow]
scalar PaltSDRE2=altSDRE2[Irow]

' quality
scalar PaltMaxQU=altMaxQU[Irow]
scalar PaltMedQU=altMinQU[Irow]
scalar PaltMinQU=altMinQU[Irow]

' ------- Slope -------
scalar PpendMaxOpt=pendMaxOpt[Irow]
scalar PpendMaxAct=pendMaxAct[Irow]
scalar PpendNoAC=pendNoAC[Irow]

' ------ Insolation --------
```

```
scalar PinsoMin=insoMin[Irow]
scalar PinsoMed=insoMed[Irow]
scalar PinsoMax=insoMax[Irow]

' ------ Road distance --------
scalar PDiEqOpt1=DiEqOpt1[Irow]
scalar PDiEqoPT2=DiEqptAD2[Irow]
scalar PDiEqSD1=DiEqSD1[Irow]
scalar PDiEqSD2=DiEqSD2[Irow]


' ============================================================
'           CREATE AND SAVE INDICATOR LAYERS
' ============================================================

' - Altitude -
import Alt.txt as(ArcGis) gen(Alt) type(float)
gen Ialta=mbell(Alt,PaltOptAD1,PaltOptAD2,PaltSDAD1,PaltSDAD2) _
          lab "altitude indicator for environmental suitability"
gen Ialtr=mbell(Alt,PaltOptRE1,PaltOptREE,PaltSDRE1,PaltSDRE2) _
          lab "altitude indicator for productivity"
gen Ialtq=msigon(Alt,PaltMaxQU,PaltMedQU,PaltMinQu) _
          lab " altitude indicator for quality"

' ===== Slope ====== from degree to % ======
import Slope_percent.txt as(ArcGis) gen(Slope) type(float)
gen Islope=msigoff(Slope,PpendMaxOpt,PpendMaxAct,PpendNonAC) _
          lab "Slope indicator"

' ==== Insolation =======
import hs75sud.txt as(ArcGis) gen(hs75sud) type(float)
gen Ihs=mSigon(hs75sud,PinsoMin,PinsoMed,PinsoMax) _
          lab "Insolation indicator"

' ==== road distance / height difference ====
import Distrade.txt as(ArcGis) gen(Distrade) type(float)
import DislivCS.txt as(ArcGis) gen(Disliv)   type(float)
gen D=sqrt(Distrade*Distrade+Disliv*Disliv)
gen Pp=Disliv/Distrade*100
drop Distrade Disliv
gen IDE=D*(1+0.1*Pp)
drop D Pp
gen Idieq=mBell(IDE,PDiEqOpt1,PDiEqOpt2,PDiEqSD1,PDiEqSD2) _
         lab "equivalent distance indicator"
drop IDE

' ======== Use of the soil =====
import UsoSuolo.txt as(ArcGis) gen(UsoSuolo) type(int)
gen Isuolo=1 if UsoSuolo=5|Ususuolo=8|Ususuolo=9|Ususuolo=28|Ususuolo=44


' ============================================================
'                    LAND SUITABILITY INDEX
' ============================================================

' ===== ENVIRONMENTAL SUITABILITY macro-indicator =============
gen IVn=(Ialta+Ihs)/2
class IVn 0.2 0.45 0.6 0.75  gen(Vn)
header varlab Vn "environmental suitability"

' ===== AGRONOMIC SUITABILITY macro-indicator =============
gen IVr=(IVn+Ialtr+Islope+Idieq)/4
class IVr 0.2 0.45 0.6 0.75  gen(Vr)
header varlab Vr "Agronomic suitability"

' ===== QUALITY macro-indicator ===========
gen IVq=(IVn+Ialtq)/2
class IVq 0.2 0.45 0.6 0.75  gen(Vq)
header varlab Vq  "Quality suitability"
```

```
'============ To fit weight ======================
use %3%                  ' loads weight set
scalar Irow=findrow(criterio,"%4%")
scalar Pr=tPr[Irow]      ' weight for productivity macro-indicator
scalar Pq=tPq[Irow]      ' weight for quality macro-indicator
scalar Pn=taPn[Irow]     ' weight for environmental indicator

gen IV=Isuolo*(Pn*IVn+Pr*IVr+Pq*IVq)
class IV 0.2 0.45 0.6 0.75  gen(Voc)
header varlab Voc  "suitability index"
export Voc as(ArcGis) saving(Voc.txt) replace
map Voc legend(2,2,100) _
        labels image(Hs75NO.bmp,50) _
        vect(Roads_and_railways_2000.shp,16512,1) _
        vect(Human settlements.shp,8421504,3) _
        vect(Villages.shp,16711680,1)
```

The meaning of the SemGrid commands used in the `HerbLand.cmf` script is available below.

Command	Description
class	creates a code variable by classifying a continuous variable
distance	creates a layer with the minimum distance values from target cell types
drop	erases variables/observation from the current dataset
encode	encodes categorical variables creating a code variable and its legend
export	exports current grid layers in different formats (ArcGis, Surfer, etc.)
generate	calculates new variables using mathematical expressions
header	lists, modifies and inserts header items and labels in the current dataset
import	imports to current project, grids of different formats ((ArcGis, Surfer, etc.)
legend	displays and modifies legends for grids and table variables
map	generates a map from a grid layer
recode	recodes the values of a code variable
save	saves the current dataset
scalar	manages scalar (numerical) ambient variables
use	loads a new dataset