

tool

```

tool.tns ist eine Sammlung nützlicher Funktionen Haftendorn 2011
tool.tns muss in mylib stehen. Bibliotheken aktualisieren
stomat(k) erzeugt stochastische nxn Zufallsmatrix
Dabei sind die Zeilensummen 1. Gebrauch siehe Extraseite
liste := {h,a,n,s,e} • {h,a,n,s,e}
tier := eldel(liste,3) • {h,a,s,e} eldel(k) löscht das k-te Element aus einer Liste,
liste bleibt unverändert! liste • {h,a,n,s,e} tier • {h,a,s,e}
lidel(liste,2,3) • {h,s,e} lidel(liste,k,r) löscht die Elemente k bis incl. r
perm(liste) • {h,n,a,e,s} perm(liste) • {a,s,h,n,e}
erzeugt eine zufällige Permutation der Liste
    
```

1.1

```

Beispiel für stochastische Matrix stomat(n) ..
aa := stomat(3) •  $\begin{bmatrix} 0.95 & 0.01 & 0.04 \\ 0.68 & 0.27 & 0.05 \\ 0.82 & 0.07 & 0.11 \end{bmatrix}$  erzeugt stochastische nxn Zufallsmatrix
Dabei sind die Zeilensummen 1 start :=  $\begin{bmatrix} 10 & 30 & 40 \end{bmatrix}$  •  $\begin{bmatrix} 10 & 30 & 40 \end{bmatrix}$ 
start-aa •  $\begin{bmatrix} 62.7 & 11. & 6.3 \end{bmatrix}$  aa20 •  $\begin{bmatrix} 0.939791 & 0.017016 & 0.043194 \\ 0.939791 & 0.017016 & 0.043194 \\ 0.939791 & 0.017016 & 0.043194 \end{bmatrix}$ 
Die end := start-aa20 •  $\begin{bmatrix} 75.1832 & 1.36126 & 3.4555 \end{bmatrix}$ 
 $\sum_{i=1}^3 (\text{start}[i,j]) \cdot 80 = \sum_{i=1}^3 (\text{end}[i,j]) \cdot 80$ . aa20 gibt die langfristige Verteilung der 80 Leute an.
    
```

1.2

```

stomat 1/20
Define LibPub stomat(n)=
Func
© stomat (n) gibt quadr. stochastische Zufalls-Matrix dim nxn aus.
Local i,j,ma,lima,h,z
ma:=newMat(n,n)
lima:={ }
For i,1,n © n Zeilen h werden erzeugt
z:=randInt(0,100) © volle Prozente, Division später
h:={z}
For j,2,n-1
While sum(h)>z-100 © Wahrsch. nicht größer als 100%
z:=randInt(0,100)
EndWhile
h:=augment(h,{z})
EndFor
z:=100-sum(h)
h:=augment(h,{z}) © Hier ist eine Zeile fertig
tool perm (h) © vermeidet, dass die großen Werte bevorzugt vorn sind
lima:=augment(lima,h) © Lange Liste aller Werte
EndFor
lima:=lima-0.01 © jetzt sind es Wahrscheinlichkeiten
    
```

1.3

```

eldel 2/2
Define LibPub eldel(liste,k)=
Func
©ldel (liste,k) gibt Liste ohne das Element k zurück
Return augment (left(liste,k-1),right(liste,dim(liste)-k))
EndFunc
    
```

1.4

```

lidel 2/2
Define LibPub lidel(liste,k,r)=
Func
©lidel (liste,k,r) gibt Liste ohne die Elemente k bis r zurück
Return augment (left(liste,k-1),right(liste,dim(liste)-r))
EndFunc
    
```

1.5

```

perm 6/1
Define LibPub perm(liste)=
Func
© perm (liste) permutiert die Liste zufällig
Local i,e,z,n,phi
n:=dim(liste)
phi:={ }
For i,1,n-1
z:=randInt(1,dim(liste))
e:=mid(liste,z,1)
phi:=augment(phi,e)
liste:=eldel(liste,z)
EndFor
Return augment(phi,liste)
EndFunc
    
```

1.6