Cobewebbing

```
#################################################################
#way to fool XPP into cobwebbing f(x) the discrete  logistic function
# first  define a function that every other step evaluates the map
# -- in the alternate steps, it just keeps the same
# value so that it alternates between horizontal and vertical jumps
# using  the function mod(2,n) which when n=2
#
#   mod(t,2) = 0     if t is even
#            = 1     if t is odd
#
#Then, define the function g(x,y)
#   g(x,y) = y       if t is odd
#          = f(x)    if t is even
#
g(x,y)=if(mod(t,2)<.5)then(f(x))else(y)
#################################################################
# note that 't' is the iteration number 0,1,2,...
# if t is even evaluate f otherwise keep the old y
#  that is now define the 2-D map that returns the points
#  (x(t),y(t)) that make up the cobweb,
#  #
y(t+1)=g(x,y)
x(t+1)=if(t==0)then(x)else(y)
#     t=0,1,2,...,total
#################################################################
# Now we define the function f(x) for the 1-D map that we iterate
#  the logistic map in this case,
# and the initial conditions.
# note that x(t+2)=f(x(t)) so every other point is the map f(x)!
# the logistic map in this case
f(x)=a*x*(1-x)
par a=2
# always start y=0
# x(0), y(0)
init x=0.1,y=0
# If you only want to draw functons, start at a fixed point
#
#some good value for the parameter a are:
#
#a=0.5 0 stable fixed point
#a=3.5  stable period 4 points, unstble period 2 points
# a=1.0  TC bifurcation of fixed points
# a=2.0  different stable fixed point
# a=3.00 PD bifurcations
# a=3.04 stable period 2 orbit
# a=3.45 period 2 orbit destabilizes
# a=4.00 periodic orbits of all periods
#
#################################################################
# Here we define some variables used to plot the lines y=x
# and the curve y=f(x0 on the same graph for the cobwebbing.
#
# the larger nit gives a smoother curve, but nit must equal total
#  the total number of iterations of the map
#
par xlo=0,xhi=1,nit=2000
xx=xlo+(xhi-xlo)*t/nit
# always start y=0
#
## if you wish to plot iterates of the map and the line y=x
aux map=f(xx)
aux st=xx
## and if you want to plot higher iterates of the function
#f(x)
```

$$\text{mod}(t,2) = \begin{cases} 0, & \text{if } t \text{ even integer} \\ 1, & \text{if } t \text{ odd integer} \end{cases}$$

$$(x(0),y(0)) \to (x(0),f(x(0))) = (x_0,x_1) \to (x_1,x_1) \to (x_1,f(x_1)) = (x_1,x_2) \to (x_2,x_2) \cdots$$

function to iterate

initial condition $(x(0),y(0)) = (0.1,0)$

To plot functions for x ranging from xlo to $(xhi-xlo)\left(\frac{\text{total}}{nit}\right)$ by joining line segments of length $(xhi-xlo)/nit$.

Can only plot t, x(t), y(t) and auxiliary variables

```
aux map2=f(f(xx))
aux map3=f(f(f(xx)))
aux map4=f(f(f(f(xx))))
#
#####################################################
# in silent mode when you only want to output y and st
#only  y,st
# some convenient settings for the graphics
@ xlo=0,ylo=0,xhi=1.001,yhi=1.001
# Setes the axes as horzotal x ad vertcal y and plots the iterates
@ xp=x,yp=y
@ nplot=5
# add the plots y=x and y=f(x)
@ xp2=st,yp2=st
@ xp3=st,yp3=map
#  if you increase nplot you can also plot higher iterates
#  i.e., the 2nd, 3rd, or 4th iterate
@ xp4=st,yp4=map2
#@ xp3=st,yp3=map3
@ xp5=st,yp5=map4
#
#########################################################
#
# tell xpp that it is a discrete map and iterate nit=total times
@ meth=discrete,total=2000
#@ output=file1.dat
#
# All xppaut code must end with "done" or "d" to
# tell its compiler where the end of the code is.
#
done
#
#####################################
```

Sets axis limits for 2D graphs

Cobwebbing

Plots 5 graphs

line y=x

y=f(x)

$y = f^2(x)$

$y = f^4(x)$

difference equation NOT differential equation

Beware: If nit and total are not large enough the graph may not accurate.