Building your own discrete model

Part 1

Aim : build a model of population dynamics, which initially shows exponential growth and eventually saturation at constant value.

We will start with a familiar model, namely, the exponential growth model

$x_t = a * x_{t-1}$

This model show exponential growth, but not the saturation to a constant value.

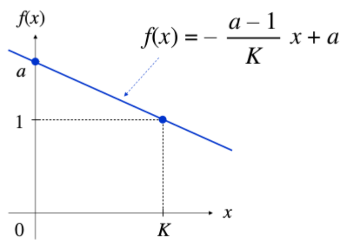
If a is 1 then the dynamics is constant. Thus we can make a a function f (x), where f (x) is some function that will tend to 1 as x becomes large.

$x_t = f[x_{t-1}]\, x_{t-1}$

Lets choose the simplest possible form of f[x] that has the desired property. The simple choice is the linear function

f[x] = -(a - 1)/K*x + a

This function jas the following property : $f[0] = a$ and $f[K] = 1$. Thus a is the initial rate of growth, when the population is close to zero, and K is the population at which the system will saturate since the rate of growth at that point is 1.



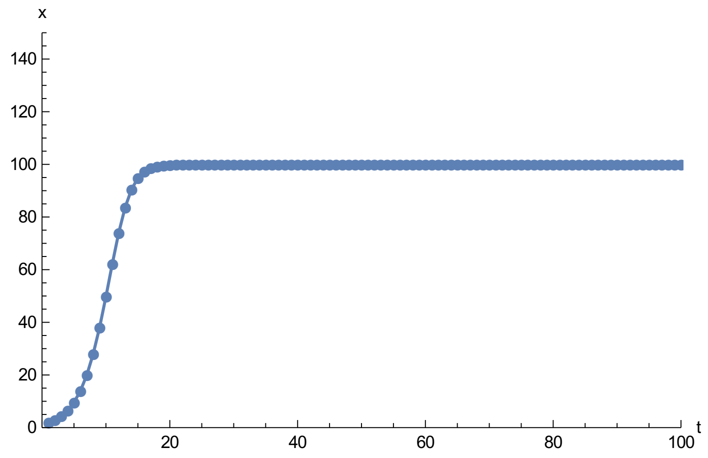
Substituting this function into the exponential growth model gives the following update rule

$x_t = (-(a-1)/K * x_{t-1} + a)\, x_{t-1}$

This model is in fact the famous logistic map.

Lets numerically simulate the model. We will use the following parameters $a = 1.5$, $K = 100$, and $x_0 = 2.0$.
i.e. The population will start at 2.0 and will saturate at 100.0. We will record the time series for 100 steps.

```
results = {};
(*initialize*)
x = 2.0;
a = 1.5;
K = 100.0;

AppendTo[results, x];

(*update loop*)
Do[x = (- (a - 1) / K * x + a) * x;
 AppendTo[results, x]; (*record the time series data*)
 , {100}]

(*Plot the results*)
ListPlot[results, Joined → True, PlotRange → {{0, 100}, {0, 150}},
 PlotMarkers → Automatic, AxesLabel → {"t", "x"}]
```

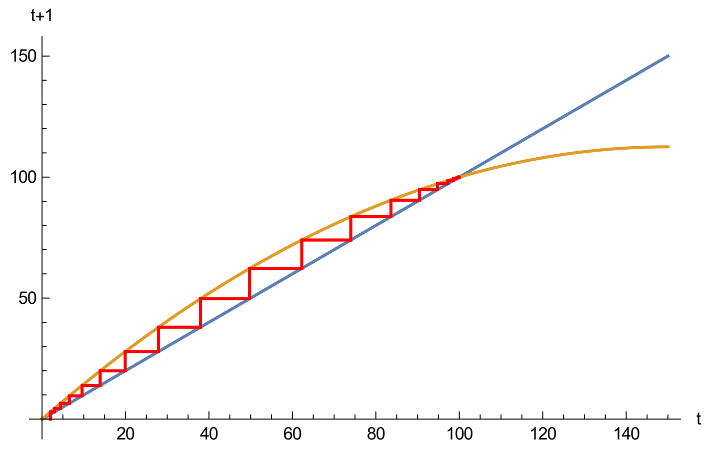


As expected we see observe an initial exponential growth and then saturation to a constant value of 100.

An interesting way of visualizing discrete one dimensional dynamical system is vai a **Cob Web plot**. The following code plots the results on the Cob Web diagram.

```
CobWebData[timeseries_] := Module[{p1, p, j},
  p = {};
  AppendTo[p, {timeseries[[1]], 0}];
  For[j = 1, j ≤ Length[timeseries] - 1, j ++,
   AppendTo[p, {timeseries[[j]], timeseries[[j + 1]]}];
   AppendTo[p, {timeseries[[j + 1]], timeseries[[j + 1]]}];
  ];
  Return[p];
 ]
```

```
Show[Plot[{x, (- (a - 1) / K * x + a) * x}, {x, 0, 150}, AxesLabel → {"t", "t+1"}],
 ListPlot[CobWebData[results], Joined → True, PlotStyle → Red]]
```

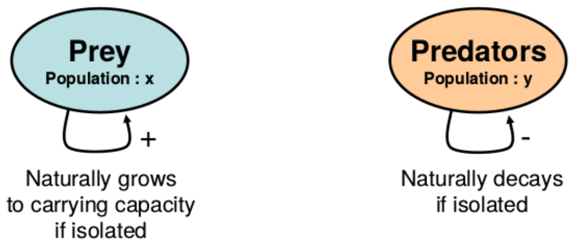


Part 2

Aim 2 : Build a model with two species : one predator and one prey

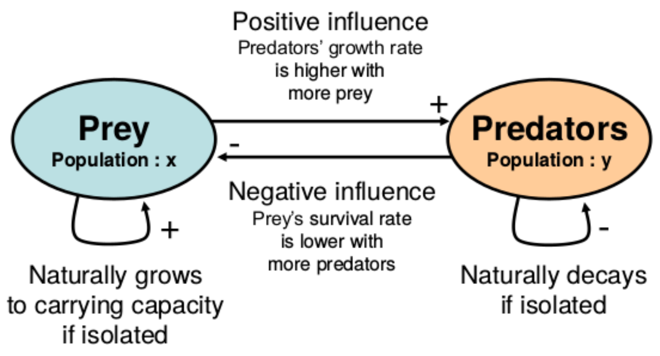
Lets denote the population of prey by x and the population of predators by y.

First note the inherent dynamics of x and y
* Prey grows if there are no predators
* Predators die if there is no prey


Prey
Population : x
+
Naturally grows to carrying capacity if isolated
Predators
Population : y
-
Naturally decays if isolated

Causal loop diagram illustrating the inherent
dynamics of each of the prey and predator population.

Now consider interactions between the two species
* Prey death rate increases as the populaiton of predators increases
* Predators growth rate increases as the prey population increases


Positive influence
Predators' growth rate is higher with more prey
+
Prey
Population : x
-
Predators
Population : y
Negative influence
Prey's survival rate is lower with more predators
+
Naturally grows to carrying capacity if isolated
-
Naturally decays if isolated

For the inherent dynamics use existing model

For prey, use logistic equation

$x_t = x_{t-1} + r_x * x_{t-1} (1 - x_{t-1} / K)$

For predators, use exponential decay

$y_t = y_{t-1} - d_y \, y_{t-1}$

Now add the interaction terms

$x_t = x_{t-1} + r * x_{t-1} (1 - x_{t-1} / K) - d_x [y_{t-1}] * x_{t-1}$

$y_t = y_{t-1} - d y_{t-1} + r_t [x_{t-1}] * y_{t-1}$

Now we need to decide, which functions to use for d_x and r_t.

The death rate of the prey should be 0 if there are no predators, while it should approach 1 (= 100 % mortality rate!) if there are too many predators.

$d_x[y] = 1 - 1/(b * y + 1)$

The growth rate of predators is proportional to the number of prey and can increase indefinitely. The simplest mathematical relation with this property is the linear function

$r_t[x] = c * x$

Check the extreme values of the functions d_x and r_t

$d_x[0] = 0$, $d_x[\infty] = 1$;
$r_t[0] = 0$, $d_t[\infty] = \infty$;
That seems reasonable

Now substituting the expressions for d_x and r_t into
our dynamics equations we obtain the following update rules

$x_t = x_{t-1} + r * x_{t-1} (1 - x_{t-1} / K) - \left(1 - 1/(b * y_{t-1} + 1)\right) * x_{t-1}$
$y_t = y_{t-1} - d * y_{t-1} + (c * x_{t-1}) * y_{t-1}$

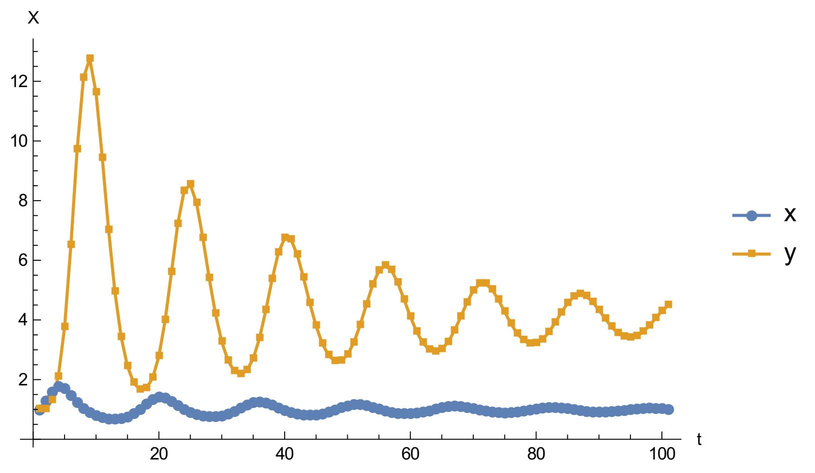
Now lets numerically simulate the dynamics using the following set of parameters
$r = b = d = c = 1$, $k = 5$;
$x_0 = y_0 = 1$

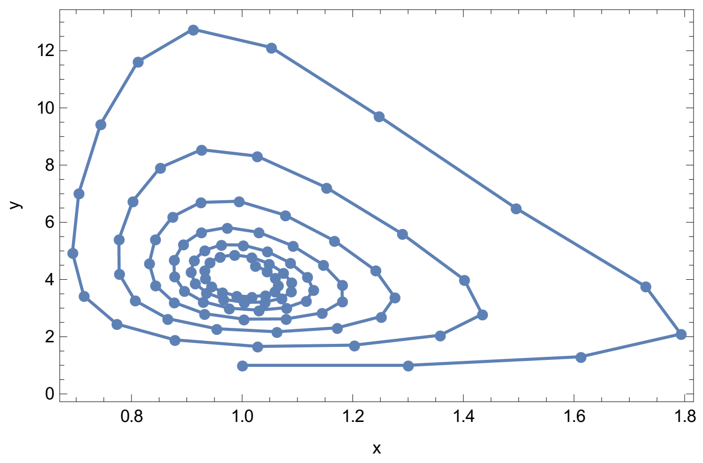
```
xnew = x + r * x * (1 - x / K) - (1 - 1/ (b * y + 1)) * x;
ynew = y - d * y + c * x * y;
```

```
results = {};
r = 1.0; b = 1.0; d = 1.0; c = 1.0;
K = 5.0;
(*initialize*)

x = 1.0;
y = 1.0;

AppendTo[results, {x, y}];

(*update loop*)
Do[xnew = x + r * x * (1 - x / K) - (1 - 1 / (b * y + 1)) * x;
 ynew = y - d * y + c * x * y;
 x = xnew;
 y = ynew;
 AppendTo[results, {x, y}]; (*record the time series data*)
 , {100}]
```

Lets generate the time series plot and the phase space plot.

```
ListPlot[{results[[All, 1]], results[[All, 2]]}, Joined → True,
 PlotMarkers → Automatic, AxesLabel → {"t", "X"}, PlotLegends → {"x", "y"}]

ListPlot[results, Joined → True,
 PlotMarkers → Automatic, Frame → True, FrameLabel → {"x", "y"}]
```





The derived model is a discrete variant of the Lotka - Volterra model for the predator prey dynamics.

Finding equilibrium point

The equilibrium point or fixed point is the point in phase space that is invariant under the application of the dynamic map i.e. $x_{eq} = F[x_{eq}]$ where F is the dynamic map.

Lets find the equilibrium point for our predator prey model. The equilibrium point is found by solving the following equations

$$x_{eq} = x_{eq} + r * x_{eq}\ (1 - x_{eq} / K) - \left(1 - 1 \Big/ (b * y_{eq} + 1)\right) * x_{eq}$$
$$y_{eq} = y_{eq} - d * y_{eq} + (c * x_{eq}) * y_{eq}$$

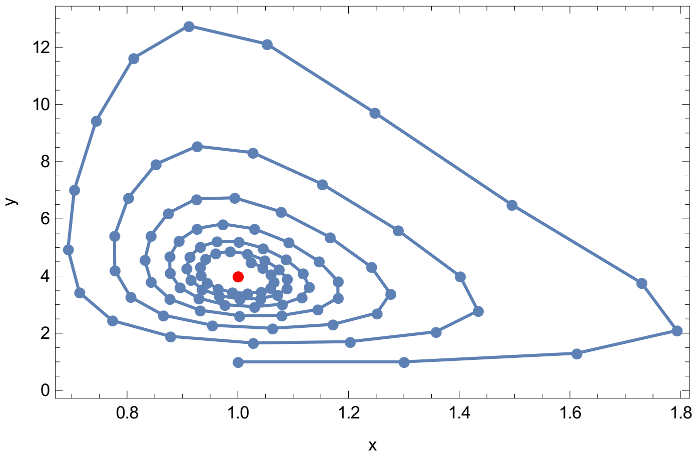
Solving this equations using the standard algebraic manipulations gives (exercise)

$$x_{eq} = c \big/ d$$
$$y_{eq} = \frac{(d - c\,K)\,r}{b * (c * K\,(-1 + r) - d\,r)}$$

```
1.

4.
```

Lets add this fixed point to our phase plot diagram

```
Show[ListPlot[results, Joined → True,
   PlotMarkers → Automatic, Frame → True, FrameLabel → {"x", "y"}],
 ListPlot[{{x_eq, y_eq}}, PlotMarkers → Automatic, PlotStyle → Red]]
```



Exercise : try several different parameters of c, b, d, r, K and the initial conditions.

Longer Exercise : develop a discrete - time mathematical model of two species competing for the same resource, and simulate its behaviour.