Math 132.9

Lesson 9: Population Models and Numerical Methods

9.1 Application.

Population models attempt to predict future population sizes. Important examples include management of epidemics and fish populations. We will consider several models, but we will direct our efforts toward the fish population model. Consider a lake which is capable of supporting a given fish population. If no fish are removed, then one can expect the fish population to level off to this maximum number. However, if the fish are harvested at a large rate (number of fish per unit time), then the fish population may decrease to an unacceptable level. We would like to be able to determine the fish population as a function of these parameters.

9.2 Math Model.

We will consider three continuous models for the population size:

birth , b, and death , d, rates are constant,

birth and death rates are not constant, b - d = k(M - y) and

birth and death rates are not constant and nonzero harvesting rate, H.

Here y(t) is the population size at any instant in time, t. The birth and death rates are the number of births or deaths in a unit of time per unit of population. So, if the unit of time is one day, then a birth rate = .01 means that .01*y(t) fish are born each day. M is the maximum population size and H is the number of fish caught in each day.

Birth and Death Rates are Constant.

The continuous model can be used for very large populations where y(t) is viewed as a continuous function of time. The change in the population size over a change in time, dt, is

$$y(t + dt) - y(t) \approx \text{births} - \text{deaths}$$

$$= dt\ b\ y(t) - dt\ d\ y(t)$$

Now the population, y(t), will change a little as time moves from t to t+dt, and this is the reason we do not have an equality sign for the above change in population. If we let dt go to zero, we just get zero on both sides! However, if we divide by dt, let dt go to zero and use the definition of a derivative, we get the differential equation

$$y' = (b - d)y$$

This differential equation with known b - d and given initial condition $y(0) = y_0$ is the continuous model.

Birth and Death Rates Vary with Population, b - d = k(M - y(t)).

The change in the population size over a change in time, dt, is

$$\begin{aligned} y(t + dt) - y(t) &\approx \text{births - deaths} \\ &= dt\ (b - d)\ y(t) \\ &= dt\ k(M - y(t))\ y(t) \end{aligned}$$

Again the population, y(t), will change a little as time moves from t to t+dt, and this is the reason we do not have an equality sign for the above change in population. If we divide by dt and let dt go to zero, we get the differential equation

$$y' = k(M - y)y$$

This differential equation with known k and M and given initial condition $y(0) = y_0$ is the continuous model. This is the **logistic differential equation** and it also is a model for the spread of information via personal communication.

Harvesting and b - d = k(M -y) Varies with Population.

The change in the population size over a change in time, dt, is

$$\begin{aligned} y(t + dt) - y(t) &\approx \text{births - deaths - harvesting} \\ &= dt\ (b - d)\ y(t) - dt\ H \\ &= dt\ k(M - y(t))\ y(t) - dt\ H \end{aligned}$$

If we divide by dt and let dt go to zero, we get the differential equation

$$y' = k(M - y)y - H$$

This differential equation with known H, k and M and given initial condition $y(0) = y_0$ is the continuous model. If the lake was stocked with new fish, then - H would be replaced by a positive stocking rate, S.

9.3 Method of Solution.

The first model where b - d are constant will only be good enough approximation for small changes in population sizes. The algebraic solution has the form

$$y(t) = y(0)\, e^{(b-d)t} \text{ where}$$

y(0) is the initial population at some starting time set equal to zero. If b-d > 0, then the population will increase to any large population size. Eventually, the environment will not be able to support this population size, and therefore, the b-d will have to decrease or even become negative. If b-d < 0 , then the population will decrease to zero.

The next two models with variable b-d have solutions which are a little more difficult to find, and they are similar to the spread of information via personal communication model. We will use three methods: the Excel macro euler, MATLAB/MAPLE CAS command dsolve, and a new MATLAB command ode45. This last command is an implementation of much more capable numerical scheme for solving differential equations. Recall in solving the spread of information models we had some difficulty with the euler macro; we needed to either choose much smaller time steps or to use a "better" numerical method such as in the eulert macro. The scheme used in the MATLAB command ode45 will also work well for these problem, and it will work well for many systems of differential equations which might arise from models with interacting populations. The ode45 is an implementation of the Runge-Kutta-Felberg variable step size method which should be described in a course on numerical analysis.

The general problem of solving or approximating the solution of the differential equation

$$y' = g(t,y)$$

can be approached via a variation of the first order finite difference method. Since y'(t) is the derivative of y(t), and it can be approximated by

$$y'(t) \approx (y(t + dt) - y(t))/dt$$

where t is any time and dt represents a small change in time. Now let t = i*dt, replace y(t) by Y(i) and replace the differential equation by the following discrete equation

$$(Y(i+1) - Y(i))/dt = g(i*dt, Y(i)).$$

Next we solve for Y(i+1) in terms of the previous Y(i). We hope the error, Y(i) - y(i*dt), will be small, and it will decrease as dt goes to zero.

Euler's Numerical Method for Approximating y' = g(t,y).

Let Y(0) be given.

Y(i+1) = Y(i) + dt*g(i*dt,Y(i)).

The Euler method is implemented in the macro euler. The following variation is implemented in the macro eulert. Inspection of this method shows that at each time step one must implicitly solve for the next value of the unknown, Y(i+1). In the macro eulert this in done by successive approximation in the inner loop where one must choose the length of this inner loop. The Euler-trapezoid scheme is generally more accurate and requires fewer time steps than the Euler method.

Euler-Trapezoid Numerical Method for Approximating y' = g(t,y).

Let Y(0) be given.

Y(i+1) = Y(i) + dt/2*(g(i*dt,Y(i) + g((i+1)*dt,Y(i+1)).

Macros gfct and eulert:

```
Function gfct(t,y)
   k=1/18
   M=10
   H=10
   gfct = k*y*(M - y)
End Function

Sub eulert()
   tmax = 10
   n = 100
   dt = tmax / n
   yo = 1
   Row = 460
   col = 1
   Dim y(300)
   y(1) = yo
   Cells(Row, col + 1).Value = y(1)
   For i = 1 To n
      t = (i - 1) * dt
      Cells(i + Row, col).Value = t + dt
      yy = y(i) + dt * gfct(t, y(i))
      For k = 1 To 6
```

```
        yy = y(i) + dt * (gfct(t, yy) + gfct(t + dt, y(i))) / 2
    Next k
    y(i + 1) = yy
    Cells(i + Row, col + 1).Value = y(i + 1)
  Next I
End Sub
```

9.3.1 Table Method via eulert

The tables that are the output of the macro eulert() can have a large number of entries. In the computations below the output was graphed, and this gives a very nice overview of all the numbers in the output table. However, very precise values in the output table cannot be determined from the graph. Here we consider the second model

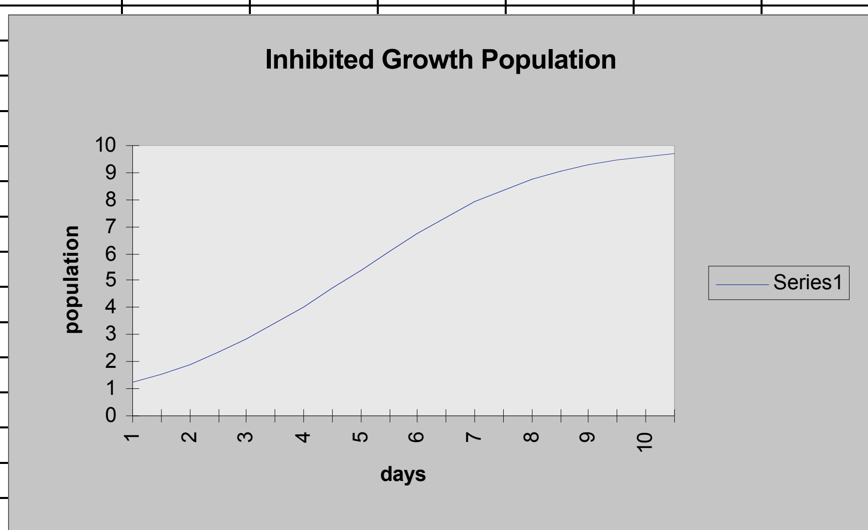
$$y' = k(M - y)y \text{ and } y(0) = yo.$$

In the calculations below we assumed the initial population was 1 unit, after 1 unit of time the population was 1.5, and the maximum population is 10. So, yo = 1, M = 10 and k is computed as follows from the Euler method where i = 0

$$(1.5 - 1.0)/1 = k*1*(10 - 1)$$

$$k = 1/18.$$

	1
1	1.25
1.5	1.553819
2	1.918371
2.5	2.349025
3	2.848256
3.5	3.41409
4	4.03867
4.5	4.707443
5	5.39951
5.5	6.089521
6	6.750992
6.5	7.36027
7	7.899968
7.5	8.360807
8	8.741501
8.5	9.047089
9	9.286563
9.5	9.470601
10	9.609871
10.5	9.714013

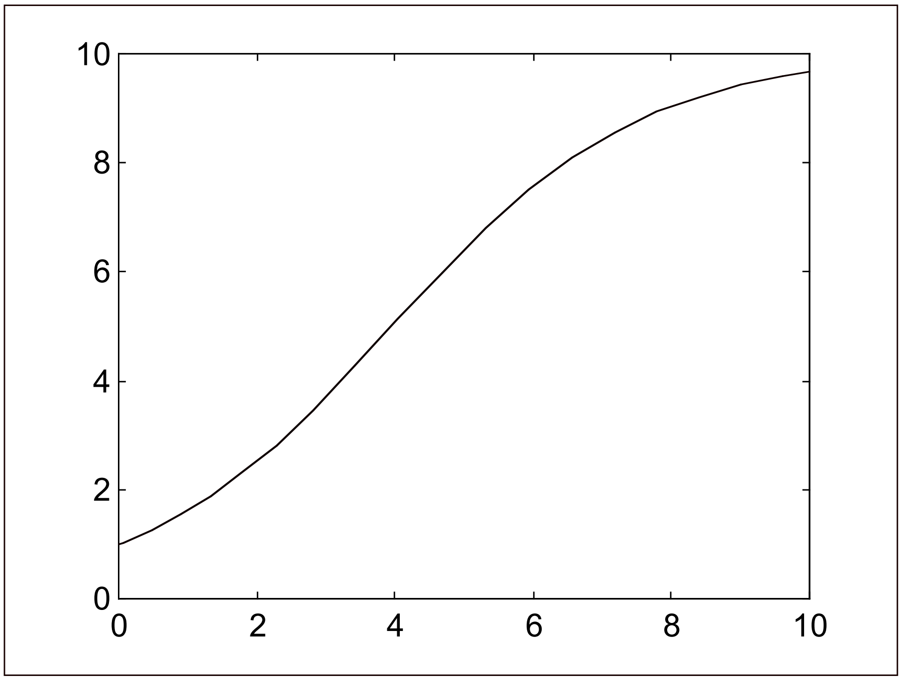


9.3.2 Graph Method via ode45.

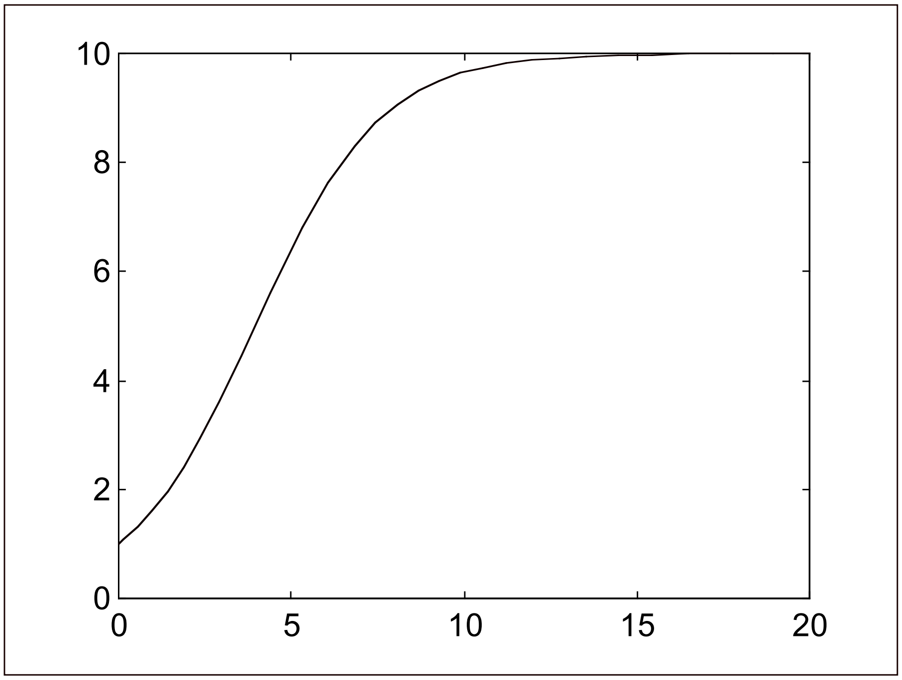
The first graph below is for the fish population with no harvesting

$$y' = (1/18)y(10 - y) \text{ and } y(0) = 1.$$

It was generated via the ode45 where the time interval was from zero to 10, and it is consistent with the output of the above macro euler. It suggests that the population will level off at M = 10, the maximum population, and the second graph confirms this is true.



y' = (1/18)y(10 - y) and y(0) = 1 for t in [1 10]



y' = (1/18)y(10 - y) and y(0) = 1 for t in [1 20]

The next two graphs are for the fish population with harvesting rate H = .2

$$y' = (1/18)y(10 - y) - .2 .$$

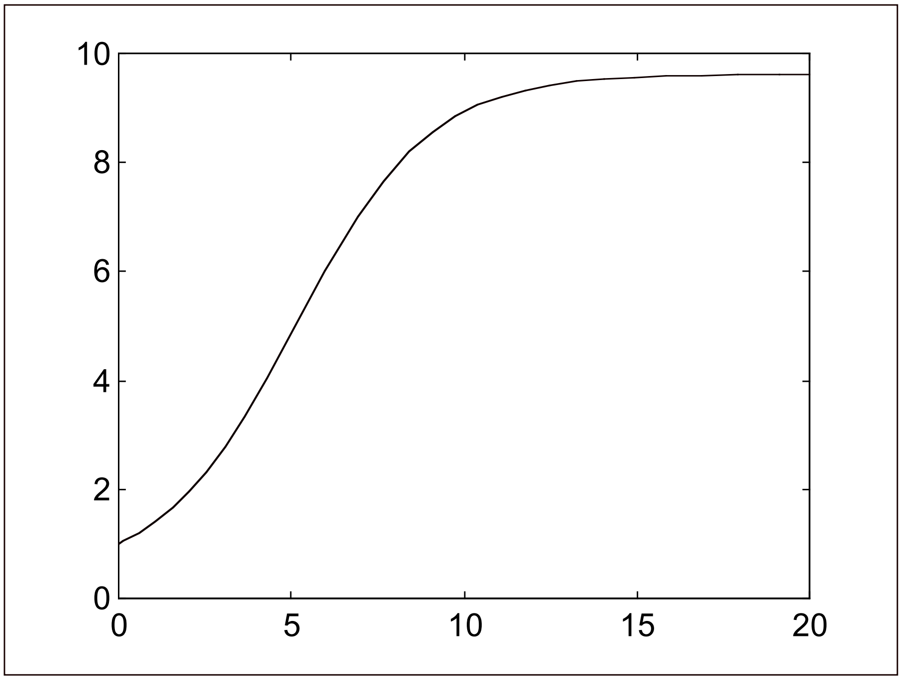
Here the **steady state solutions**, also called the **equilibrium populations**, are defined by setting the right side equal to zero and solving for y:

$$(1/18)y(10 - y) - .2 = 0$$

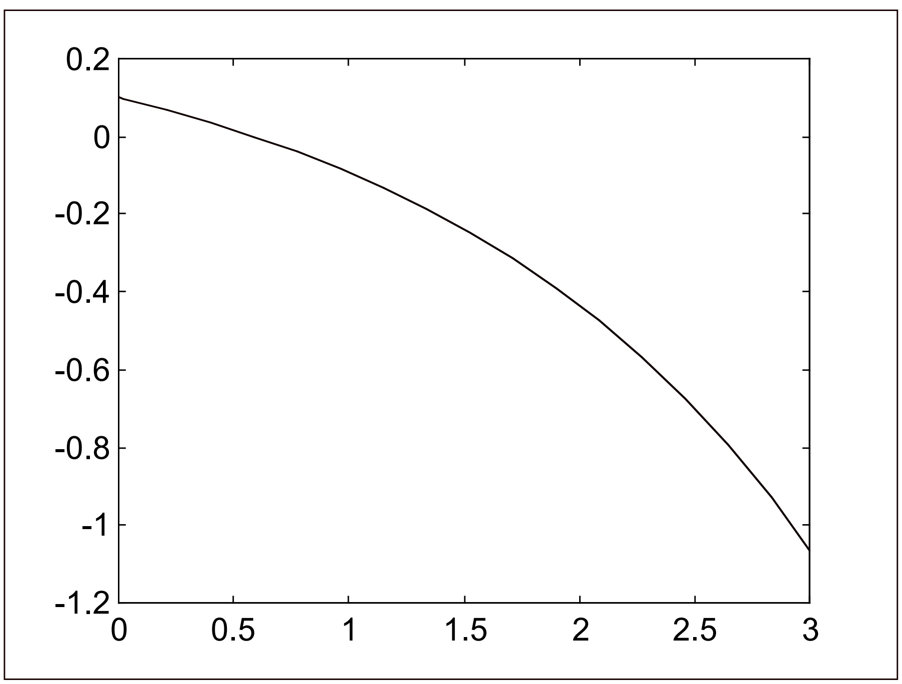
$$y^2 - 10\,y + 3.6 = 0$$

So, $y = (10 \pm \quad 85.4)/2 = .374$ and 9.626.

If the initial population is large enough, the fish population will increase to the larger steady state solution. This is illustrated in the next graph where the initial population is one. However, if the initial fish population is too small, the fish population will go to zero. This is illustrated in the second graph where the initial population is .1 units. Here the fish population drops to zero in about t = .5 units of time.



y' = (1/18)y(10 - y) - .2 and y(0) = 1



y' = (1/18)y(10 - y) - .2 and y(0) = .1

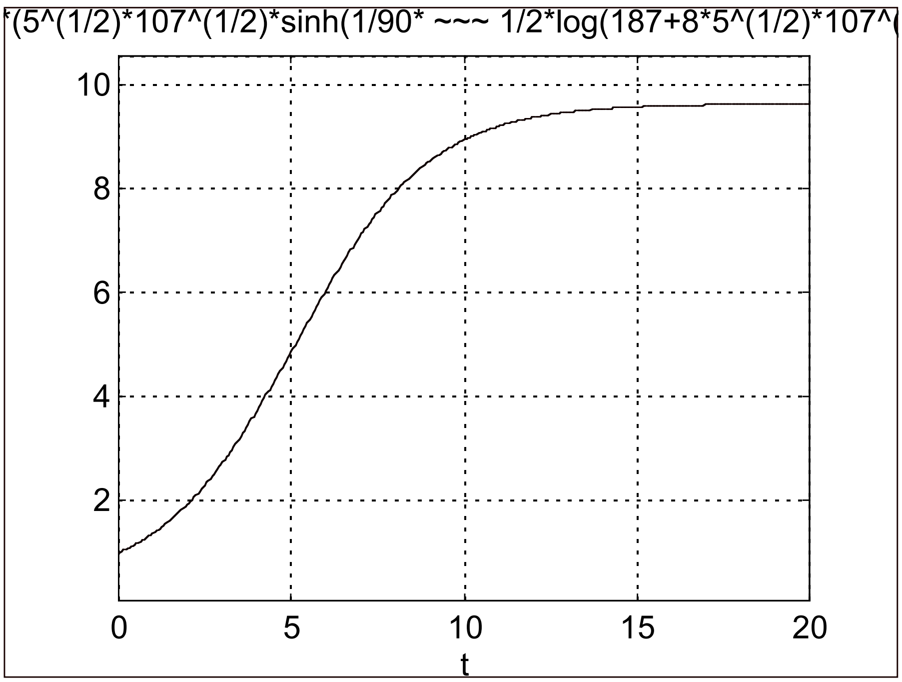
9.3.3 Algebra Method via dsolve.

The following commands use dsolve and ezplot to find the algebraic solution and graph to the fish population with harvesting rate H = .2

$$y' = (1/18)y(10 - y) - .2 \text{ and } y(0) = 1.$$

```
EDU» dsolve('Dy = (1/18)*y*(10 - y) - (2/10)', 'y(0)=1','t')
ans =
1/5*(tanh(1/9630*(107*t-9*log(-(535^(1/2)+20)/(-
535^(1/2)+20))*535^(1/2))*535^(1/2))+5/107*535^(1/2))*535^(1/2)

EDU» ezplot(ans,[0 20])
```



9.3.1 Algebra Method with Maple

As before, we can use Maple to solve the problems involved with this worksheet. We have already seen how to use Maple's version of dsolve when we looked at the cooling problem. Here we review it again. Suppose we seed a lake with 100 fish (so y(0) = 100). The lake can sustain at most 150,000 fish (so M = 150,000), and that the growth rate of this species of fish in this environment is r = b – d = 0.2301. We do the following. Creating an exponential model of the data is easy enough; we need to solve the differential equation y' = 0.2301 y:

```
[> dsolve({diff(y(t),t)=0.2301*y(t),y(0)=100});
```

Now define a population function P that reflects the solution:

[> Pexp:=t->100*exp(0.2301*t);

Once we have this function, it is a snap to find the population at times t=5, t=10, or t=30:

[> Pexp(5); Pexp(10); Pexp(30);
315.9772401
998.4161626
99525.60095

Notice that the fish population has very quickly exceeded the maximum population the lake can actually sustain. So the exponential model is certainly bad in the short-term.

Now we consider the logistic model. Observe that in the logistic model, we are given an **equilibrium population:** if the population reaches the maximum sustainable (M), it will grow no further. Using dsolve() we can find and define an equation for the fish population that takes this fact into account. For k, we use the relationship

$$k = r / M$$

So we need to solve the differential equation

$$y' = (0.2301 / 100{,}000)\,(100{,}000 - y)\, y$$

With Maple, we accomplish this via

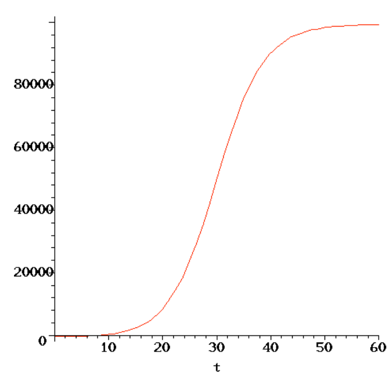
[> dsolve({diff(y(t),t)=0.2301/100000*(100000-y(t))*y(t),y(0)=100});

We can create a function to represent the result:

[> Plog:=t->t->100000/(1+999*exp(-0.2301*t));

If we plot this result over 60 days, we see how the fish population levels off just under 100,000:

[> plot(Plog(t),t=0..60);



On the other hand, let's consider what happens if we allow harvesting. A parallel to harvesting in the case of our fish-populated lake would be if we allow fishing in our lake. This gives us a harvesting constant, say H=15, and we want to know with how many fish we need to see the lake in order to prevent the population's dying out. The differential equation is now that of the harvesting model:

$$y' = (0.2301 / 100{,}000)\,(100{,}000 - y)\,y - 15$$

Our first question becomes: what are the **equilibrium populations**? Because there is harvesting, we expect that there must be a minimum population that must be maintained; if the population falls below that critical number, it will be overfished into oblivion. On the other hand, there should still be a maximum sustainable population.

To solve this, we use this fact: ***at the equilibrium population, the population y will not change; hence y' = 0*** (since y' represents the instantaneous change in population). If we substitute this into the differential equation, we get a quadratic equation that is not too hard to solve by hand:

$$0 = (0.2301 / 100{,}000)\,(100{,}000 - y)\,y - 15$$

But, why solve by hand when we can use Maple?

```
[> solve(0=(0.2301/100000)*(100000-y)*y-15);
                    99934.76840, 65.23159986
```

So the initial population of fish must be at least 65 if we want to maintain fish in the lake, and because of harvesting, the population will never exceed 99,934 (notice that, with harvesting, the maximum sustainable population is *less* than it would be without harvesting).

We can see these different behaviors more explicitly using Maple's DEplot() command. DEplot() will graph a differential equation for us. The format is similar, but not identical, to the format of the dsolve() command. We will use DEplot() to observe the behavior of the fish population when we start with 70 fish (which is enough to sustain the fish population) and when we start with 60 fish (which is not enough to sustain the fish population).

The DEplot() function is part of an external **module** to Maple, so we have to load in that module before we invoke the DEplot() command. The module we want is called DEtools.[1] To load it in, type:

```
[> with(DEtools):
```

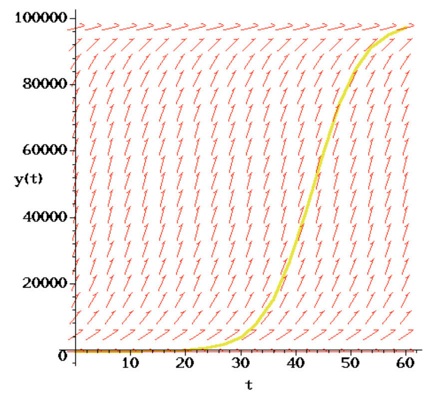
The format of the DEplot() command is

DEplot(*differential equation* , *dependent variable* ,
range for indep. vars. , [[*initial condition*]]);

so we type

[> DEplot(diff(y(t),t)=0.2301/100000*(100000-y(t))*y(t) – 15 ,y(t),t=0..60,[[y(0)=70]]);

and we obtain the following graph:

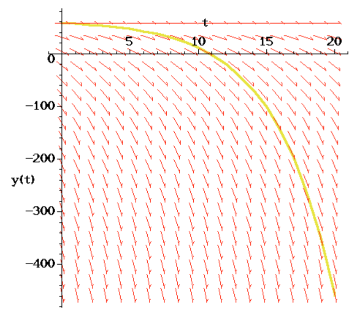


Notice that it takes the fish population longer to approach its maximum.

If we want to see what happens with an initial population of 60 (which, again, is too small), we should plot over a smaller interval of *t*-values:

[>DEplot(diff(y(t),t)=0.2301/100000*(100000 – y(t))*y(t),y(t),t=0..20,[[y(0)=60]]);

and we obtain this graph instead:



Notice that in this case, we expect the population will die off after about 11 days.

9.4 Implementation.

Instructions for Numerical Solution of y' = (1/18)y(10 – y), y(0) = 1 via the Macro eulert():

---

[1] Notice that we type a **colon** (:) instead of the usual **semicolon** (;) at the end of the subsequent with() command. One can type the usual semicolon, but Maple will give you a lot of useless information. Typing the colon suppresses output.

Step 1. Examine the macro eulert() and enter the macro via the tools, macro menu via the Visual Basic editor..

Step 2. Let tmax = 20, n = 40, yo = 1 and (row,col) = (1,1).

Step 3. Enter the function g(t,y) by using the Visual Basic editor.

Step 4. Return to a new sheet where you want the output.

Step 5. Execute the macro eulert() by via the tools, and macro menu and double click on eulert().

Step 6. Chart the output data and experiment with n. Observe "convergence."

9.5 Assessment.

The "constants" k, M and H in our two models may be difficult to determined and could even vary with time; for example, M or H might depend on the time of year. Another problem is that interaction with other populations can play a very significant role. There may be more than one fish population, and one may prey on the others so that the death rates vary with predators population size.

The differential equation y' = cy(M - y) can be solved exactly, but it does require a little more effort. Generally, numerical methods can be applied to a larger class of differential equations than algebraic methods. In the next lesson we will see how we can use the MATLAB command ode45 to solve predator-prey systems of differential equations.

9.6 Possible Homework.

1. Verify the calculations in this lesson.

2. Consider the following model with different initial conditions:

$$y' = (1/20)\, y(20 - y) - 1 \text{ and } y(0) = 1, 4, 15 \text{ and } 20.$$

   (a). Find the steady state solutions.

   (b). Use eulert to numerically solve these problems.

   (c). Use the chart wizard to graph the solutions.

3. Consider problem two with 1/20 replace by .02, .05 and .08. Compare the solutions.