Class 4: Wednesday September 11

Implementation of Algorithms on the TI-83

Implementing Euler's Method

Suppose you want to produce a table of values produced by Euler's method for the IVP:

$$P' = 0.017P \text{ where } P(0) = 100$$

with a step size of $\Delta t = 0.5$.

n	t_n	P_n	P'_n
0	0	100	1.7
1	0.5	100.85	1.7145
2	1	101.71	etc

where

$$\begin{aligned} t_n &= t_{n-1} + \Delta t \\ P_n &= P_{n-1} + \left(0.017P_{n-1}\right) \cdot \Delta t \\ P'_n &= 0.017P_n \end{aligned}$$

The following steps will produce the above table on your TI-83. But a note of *CAUTION*: your table will be worthless unless you can construct Euler's recursive equation correctly.

- Store the value of Δt. This allows you to change the value easily to see the effects of different time step size. Unfortunately, Δ is not readily available on the TI, so we'll use D.

  0.5 STO→ ALPHA D

- Set the calculator mode for sequence graphing.

  MODE use arrow keys to highlight Seq ENTER QUIT

  FORMAT highlight Time ENTER QUIT

- Go to "Y=" to define the recursive equations for t_n ($u(n)$), P_n ($v(n)$), and P'_n ($w(n)$).

  The lower case letters for u, v, w are yellow keys (above 7,8,9 keys) and the recursive n key is a combination key labelled X, T, Θ, n. The initial values t_0 ($u(n\text{Min})$), P_0 ($v(n\text{Min})$) and P'_0 ($w(n\text{Min})$) should also be set appropriately. nMin should be set to 0.

  - $u(n) = u(n-1) + D$
  - $u(n\text{Min}) = 0$
  - $v(n) = v(n-1) + w(n-1) * D$
  - $v(n\text{Min}) = 1400$
  - $w(n) = 0.017 * v$
  - $w(n\text{Min}) = 1.7$

  Note: Using the term v in the definition of $w(n)$ permits the TI to read the present value of $v(n)$.

- **Make sure you have values in TBLSET set to 1**

- Go to "Table" (a yellow key) to get a table of values. (There will be some delay as the calculator works on this). Up and Down arrow keys let you scroll through the values and the highlighted value will be given to greater accuracy at the bottom of the screen. To see $w(n)$, you need to press the right arrow key once or twice. (There will be some delay as the calculator works on this.)

Successive Approximations to Find Square Roots — The Babylonian Algorithm.
Pretend you don't have a square root key on your calculator. (The algorithm given here is quite ancient.) How can we approximate $\sqrt{2}$?
Suppose

$$\begin{aligned} a &= \sqrt{2}. \text{ Square both sides.} \\ a^2 &= 2 \text{ Divide both sides by } x. \\ a &= 2/a \end{aligned}$$

Only the true square root of 2 satisfies $\sqrt{2} = 2/\sqrt{2}$.

If x is an estimate which is less than the true value for $\sqrt{2}$ then $2/x$ is an estimate which is _____ than the true value.

If x is an estimate which is greater than the true value for $\sqrt{2}$ then $2/x$ is an estimate which is _____ than the true value.

Hence a better estimate will be _____.

General Babylonian Algorithm.

STEP 1: Let a_0 be your initial estimate for $\sqrt{r}$.

STEP 2: Then the next estimate is the average of your most recent estimate and r divided by your most recent estimate.

$$a_{\text{new}} = \frac{a_{\text{old}} + \dfrac{r}{a_{\text{old}}}}{2}$$

STEP 3: Continue calculating terms in the sequence until you reach the level of accuracy desired.

Implementing the Babylonian Algorithm

The Babylonian Algorithm to approximate $\sqrt{A}$ is

$$x_{\text{new}} = \frac{x_{\text{old}} + \dfrac{A}{x_{\text{old}}}}{2}$$

- Let's Store the value of A 2 STO→ ALPHA A
- Make sure the calculator is in sequence mode by
  MODE use arrow keys to highlight Seq ENTER QUIT
  FORMAT highlight Time ENTER QUIT
- Go to the "Y=" key to define the recursive equation for the algorithm

  Write down the equations you have to type into the calculator to implement the babylonian algorithm.