CSC160S: Intro to Scientific, Graphical and Symbolic Computing
Department of Computer Science
University of Toronto

Instructor: E. Fiume **Term Test 1** **14 Feb. 1992**

1. [5] Reparameterise the polynomial $p(x) = x(x - 1)$ from the interval $[-1,2]$ to a polynomial $\hat{p}(t)$ defined on the interval [0,1]. Express $\hat{p}$ in the power-basis (i.e., in the form $a_0 + a_1 t + a_2 t^2 + a_3 t^3 + \cdots$ (you decide on the correct degree of the polynomial).

2. [15] Remember our old friend the trochoid? Well, it so happens that the Canadian military might be willing to pay us big bucks if you can tell them interesting things about it and if you can render it quickly (I take a big commission). Recall that it was defined parametrically as follows: for arbitrary constants a and r,

$$x(u) = au - r\sin u$$
$$y(u) = r\cos u.$$

(a) [3] Describe and then sketch a plot of the trochoid in the (x,y) plane if a is set to 0, r is a nonzero constant, and $0 \le u < 2\pi$.

(b) [3] Describe and then sketch a plot of the trochoid in the (x,y) plane if r is set to 0, a is a nonzero constant, and $0 \le u < 2\pi$.

(c) [9] Here is a brute-force Turing program that renders the trochoid for nonzero values of a and r.

```
var xn,yn: int
var x,y, u, r, a, du: real
r  := 100
a  := 25
du := 0.01
u  := 0
x  := 0       % initial value of (x,y) for u=0
y  := r
loop
     xn := round(x)
     yn := round(y)
     drawdot(xn,150+yn,1)
     u  += du
     exit when u > 6π
     x  := a*u - r*sin(u)
     y  := r*cos(u)
end loop
```

Your assignment is to transform this implementation into an incremental algorithm so that

$$x_{i+1} := x_i + \frac{dx}{du}\Delta u$$
$$y_{i+1} := y_i + \frac{dy}{du}\Delta u$$

Derive the appropriate derivatives and rewrite them so that *no* sine or cosine function appears anywhere in the result. (Hint: write the derivatives in terms of previous values of x and y.) Then transform the inner loop in the above code to reflect your efficient derivation.

3. [6] A cubic Hermite curve is defined by specifying two points P_0, P_1 and two tangents T_0, T_1. The resulting curve segment $p(t)$ satisfies the following constraints:

$$\begin{aligned} p(0) &= P_0 \\ p(1) &= P_1 \\ p'(0) &= T_0 \\ p'(1) &= T_1 \end{aligned}$$

The notation p' denotes the derivative of p with respect to t. Assuming we already have a change of basis matrix M that takes us from points and tangents as above to a cubic polynomial, your goal is to show how to use this form in a piecewise cubic fashion. We would like to have a sequence $p_0, p_1, \cdots p_m$ of Hermite cubic curve segments that interpolate some sample data consisting of points and tangents. How should the sample data be arranged to make use of Hermite interpolation, and how should curve segments be stitched together so as to effect smooth interpolation of the sample data? Outline an (4 or 5 line) algorithm that performs piecewise-cubic Hermite interpolation on the data in the form you specified.

4. [4] Suppose we have a table T of values for a function $f(x)$ defined for equally spaced values of x. Thus the table contains the entries $f(0)$, $f(\Delta x)$, $f(2\Delta x)$, $\cdots$, and in fact the i^{th} entry of T, $T(i) = f(i\Delta x)$. Suppose we would like to estimate the value of $f(x)$ for some x lying between x_i and x_{i+1} for some i. Show how to use linear interpolation on the appropriate entries of T to get an estimate for $f(x)$.