THE TI-Nspire PROGRAMS

JAMES KEESLING

The purpose of this document is to list and document the programs that will be used in this class. For each program there is a screen shot containing an example and a listing of the TN-Nspire CX CAS program. The student is responsible to enter each program and be familiar with its use.

1. Solving $f(x) = 0$

In this section are the Newton-Raphson method and the Bisection method.

![Newton-Raphson Method Example](image)

**Figure 1.** The Newton-Raphson method applied to $x^2 - 2 = 0$
2. LAGRANGE POLYNOMIALS

In this section are the programs for the VanderMonde matrix and the Lagrange polynomial through a set of points \( \{(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)\} \). Note that the polynomial program calls for the points in the form \([x_0, x_1, \ldots, x_n]\) and \([y_0, y_1, \ldots, y_n]\).
Figure 3. Program for the VanderMonde matrix for the points \( \{1, 2, 3\} \)

Figure 4. The Lagrange polynomial through the points \( \{(0, 1), (1, 0), (3, 2)\} \)
3. Numerical Integration

In this section are programs to compute the normalized Newton-Cotes coefficients and to estimate an integral using Newton-Cotes integration. We also have programs for Romberg integration and Gaussian integration. These last programs are greatly superior to Newton-Cotes. For Gaussian quadrature we theoretically need the Legendre polynomials. We give a program that will list the first $n + 1$ of these polynomials from degree 0 to degree $n$. 
Figure 6. Program for the Newton-Cotes coefficients

Figure 7. Program for Romberg Integration
Figure 8. Program for Gaussian quadrature

```plaintext
Figure 9. Program for the Legendre polynomials up to degree n
```
Numerical Differentiation

In this section we give the programs needed for numerical differentiation of a function \( f(x) \). There are two of these programs. The first program determines the coefficients to be used in estimating \( f^{(k)}(a) \) using the \( n+1 \) points \( \{a - m_0 \cdot h, a - m_1 \cdot h, \ldots, a - m_n \cdot h\} \). In the programs \( b = [m_0, m_1, \ldots, m_n] \)
Figure 11. Program determining the coefficients to be used in estimating the $k$th-derivative at $a$ using the points $a - h \cdot b$ with $b = [m_0, m_1, \ldots, m_n]$.

```plaintext
define numdiff(x, h, b) begin
  dim(a[2], 1 -> m)
  numdiff[3, 4] 0 -> p
  for i,0,m
    p = coef[i+1,k]-f(x = a[i+1] - h - p)
  endfor
  disp "p = p"
end
```

Figure 12. Program giving the formula to estimate $f^{(k)}(a)$ using the points $a - h \cdot b$ with $b = [m_0, m_1, \ldots, m_n]$.

```plaintext
define numvadiff(x, h, a) begin
  dim(a[2] - 1 -> n)
  newMat(1, n+1) -> vander
  newMat(n+1, 1) -> coef
  newMat(1, n) -> temp
  vander = temp[1,1]
  for i, 1,n
    (vander-1, i+1) -> temp
    disp coef
  endfor
end
```
5. Differential Equations

In this section we give some programs useful for solving ordinary differential equations. We give a theoretical solution based on Picard iteration and numerical methods based some method of integration. We also give a program for the Taylor method.

**Figure 13.** Program giving the Picard iteration method
**Figure 14.** Example using the Taylor method

**Figure 15.** Program for the Taylor method
Figure 16. Program giving the Euler method

```
define eulermethod(s,a,b,n) -> soln
    newMat(a,1,2) -> soln
    approx[1] = x
    approx[2] = x + h
    a = soln[1,1]
    b = soln[1,2]
    for i, in
        x = x + f1(x, y)
        y = y + h
        a = soln[i,1]
        b = soln[i,2]
    endfor
    disp soln
end
```

Figure 17. Program giving the modified Euler method

```
define modified eulermethod(s,a,b,n) -> soln
    approx[1] = x
    approx[2] = x + h
    approx[3] = x + 2h
    newMat(a,2,2) -> soln
    a = soln[1,1]
    b = soln[1,2]
    for i, in
        x = x + f1(x, y)
        y = y + h
        a = soln[i,1]
        b = soln[i,2]
    endfor
    disp soln
end
```
Figure 18. Program giving the Heun method

Figure 19. Program giving the Runge-Kutta method
6. Stochastic Simulation

In this section we do not show copies of the programs. They can now be downloaded. So, there is no need for them to be given to be copied into your calculator. However, we do give examples of how the data is to be entered when the programs are run and what the typical output will look like and how it should be interpreted.

Figure 20. Program giving an example of the Bowling program. The output is a sequence of scores for a bowler whose probability of a strike, spare, and open frame are the numbers entered in that order.
Figure 21. Program giving an example of the Queue simulation program. The output gives the time spent in each state given the arrival rate, the service rate, and the number of servers.
Figure 22. This program estimates the integral \( \int_{a}^{b} dx \) by the Monte Carlo method using \( n \) random points in the interval \([a, b]\). It gives \( k \) estimates of the integral.