## Exercise Set for Math/CS 4414

1. Download the `X.mat` data file from the course web page on Canvas. Then load this data file into Matlab which produce a 1D vector $X$ in Matlab whose entries are in the interval $[0, 1]$.

   (a) Use the data in $X$ to produce two vectors $X_1$ and $X_2$ such entries of $X_1$ are in $[0, 0.5]$ but entries of $X_2$ are in $[0.5, 1]$. Entries of $X_1$ and $X_2$ should be in the increasing order. Present your results in a table as follows:

| $dim(X_1)$ | $dim(X_2)$ | $k_1$ | $k_2$ |
|---|---|---|---|
| | | | |

   where $k_1$ and $k_2$ are such that

$$X_1(k_1) = 0.415300351493410, \quad X_2(k_2) = 0.586233173298685.$$

   (b) Consider the following function:

$$f(x) = \begin{cases} x\sqrt{(a-x)/(a+x)}, & \text{for } x \in [0, 0.5], \\ x\sqrt{(x-a)/(a+x)}, & \text{for } x \in [0.5, 1], \end{cases}$$

   where $a = 0.5$. Implement this function as a user-defined Matlab function with the following interface:

```
function f = strophoids(x)
```

   Note that

| $f(X_1(211))$ | $f(X_2(211))$ |
|---|---|
| 7.832815041429145e-02 | 1.736044115012921e-01 |

   The values in the 2nd row can be used to debug your function and check whether your $X_1$ and $X_2$ vectors are formed correctly. Then demonstrate how your function works by filling the following table:

| $f(X_1(511))$ | $f(X_2(511))$ |
|---|---|
| | |

   (c) Make a plot of $f(x)$ above by evaluating $f(x)$ at those values contained in $X_1$ and $X_2$.

2. Consider $\mathbb{F} = \mathbb{F}(\beta, t, L, U)$.

   (a) Show that $x_{min} = \beta^{L-1}$.

   (b) Show that $x_{max} = \beta^U(1 - \beta^{-t})$.

   (c) Show that $x^+_{min} = \big(1/\beta + 1/\beta^t\big)\beta^L$.

   (d) Show that $x^-_{max} = \beta^U\big(1 - 2\beta^{-t}\big)$.

3. In Matlab, how many machine numbers are there in $[1/2, 1]$?

4. In Matlab, $y = 1$ is a machine number. Find the machine number $x$ such that $x < y$ but there is no machine number between $x$ and $y$.

5. Consider the set of floating point numbers in Matlab $\mathbb{F} = \mathbb{F}(2, 53, -1021, 1024)$.

   (a) Use Mathematica to find an approximation to $x_{min}$. Present the result with 25 digits.

   (b) Use Mathematica to find an approximation to $x_{max}$. Present the result with 25 digits.

   (c) Use Mathematica to find an approximation to $x^+_{min}$. Present the result with 25 digits.

   (d) Use Mathematica to find an approximation to $x^-_{max}$. Present the result with 25 digits.

6. Consider the set of floating point numbers in Matlab $\mathbb{F} = \mathbb{F}(2, 53, -1021, 1024)$.

   (a) Consider $x^+_{min}$ as an approximation to $x_{min}$, find its absolution error in Mathematica. Present the result with 25 digits.

   (b) Consider $x^-_{max}$ as an approximation to $x_{max}$, find its absolution error in Mathematica. Present the result with 25 digits.

7. Let $x = 1/2$. Find $y \in \mathbb{F}(2, 53, -1021, 1024)$ such that $\mathbb{F}(2, 53, -1021, 1024) \cap (x, y) = \emptyset$, i.e., for any $z \in (x, y)$, we must have $z \notin \mathbb{F}(2, 53, -1021, 1024)$.

8. Explain why the loss of significance can happen when computing $x/y$ for a nonzero $y$ that is close to 0.

9. Download the data files `I_512_1.mat`, `J_512_1.mat`, and `V_512_1.mat` from our course web page on Canvas and load them into Matlab. This will generate three arrays `I, J, V` in Matlab. Let $S$ be the sparse matrix generated by the arrays $I, J$ and $V$, and let $S_f$ be the full matrix corresponding to $S$. Present features of the sparse matrix $S$ by filling the following table:

| | |
|---|---|
| The size of $V$ | |
| The size of $S$ | |
| The number nonzero entries in $S$ | |
| The ratio of nonzero entries and all entries of $S$ | |
| The memory used by $V$ (in megabytes) | |
| The memory used by $S$ (in megabytes) | |
| The memory needed for $S_f$ (in megabytes) | |
| The value of $S(77690, 3170)$ | |
| The value of $S(53700, 53187)$ | |

Can you generate $S_f$ in your computer?

10. Consider solving for $B\mathbf{x} = \mathbf{b}$ with $B = -A$ where matrix $A$ is given on page 142 of our textbook and

$$\mathbf{b} = [5, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0]^T$$

**Remark**: Make sure that the matrix $A$ is input correctly. One way to very it is to reproduce the vector $\mathbf{p}$ on page 152 of our textbook by Matalb command: `p = A\b` with the vector $\mathbf{b}$ given on page 142.

(a) Use the $LU$ factorization to solve $B\mathbf{x} = \mathbf{b}$ and present your results with the following table:

| | |
|---|---|
| $l_{3,1}$ | |
| $u_{2,5}$ | |
| $y_1$ | |
| $x_2$ | |

(b) Use the $LU$ factorization with partial pivoting to solve $B\mathbf{x} = \mathbf{b}$ and present your results with the following table:

| | |
|---|---|
| $l_{4,3}$ | |
| $u_{3,5}$ | |
| $p_{3,2}$ | |
| $y_2$ | |
| $x_3$ | |

(c) Use the $LU$ factorization with total pivoting to solve $B\mathbf{x} = \mathbf{b}$ and present your results with the following table:

| | |
|---|---|
| $l_{7,5}$ | |
| $u_{8,9}$ | |
| $p_{6,3}$ | |
| $q_{5,7}$ | |
| $y_4$ | |
| $x_5$ | |

(d) Use the Cholesky factorization to solve $B\mathbf{x} = \mathbf{b}$ and present your results with the following table:

| | |
|---|---|
| $l_{9,6}$ | |
| $y_5$ | |
| $x_6$ | |

11. Consider a linear system $A\mathbf{x} = \mathbf{b}$ in which $A$ is the $5 \times 5$ Hilbert matrix and $\mathbf{b}$ is such that $\mathbf{x} = [1, 1, 1, 1, 1]^T$ is the solution to $A\mathbf{x} = \mathbf{b}$.

(a) Compute $\mathbf{x}^{(5)}$ by the PCG method with the preconditioner $P = I$ for this linear system. Starting the iteration from $\mathbf{x}^{(0)} = \mathbf{0}$. Present your solution by filling the following table:

| $x_1^{(5)}$ | $x_4^{(5)}$ |
|---|---|
| | |

where $x_1^{(5)}$ and $x_4^{(5)}$ are the 1st and the 4th entries of $\mathbf{x}^{(5)}$, respectively.

(b) Is $\mathbf{x}^{(5)} = \mathbf{x}$ true? Is $\mathbf{x}^{(5)}$ a good approximation to $\mathbf{x}$?

(c) Starting the iteration from $\mathbf{x}^{(0)} = \mathbf{0}$, compute $\mathbf{x}^{(k)}$ by the PCG method with the preconditioner $P = I$ for this linear system such

that $\mathbf{x}^{(k)}$ is best possible approximation to $\mathbf{x} = [1, 1, 1, 1, 1]^T$. Present you numerical results in the following table:

| $k$ | |
|---|---|
| $x_1^{(k)}$ | |
| $x_2^{(k)}$ | |
| $x_3^{(k)}$ | |
| $x_4^{(k)}$ | |
| $x_5^{(k)}$ | |

12. Consider the same linear system $B\mathbf{x} = \mathbf{b}$ described in Problem 10.

(a) Convert the matrix $B$ to the matrix $B\_s$ in sparse-array format and present your result in the following form:

| | |
|---|---|
| The value of $I(7)$ | |
| The value of $J(7)$ | |
| The value of $V(7)$ | |
| The value of $B\_s(5, 2)$ | |
| The number of nonzero entries in $B$ | |
| Memory used for $B$ (in bytes) | |
| Memory used for $B\_s$ (in bytes) | |

(b) Compute both the Cholesky factorization $L$ and the incomplete Cholesky factorization $L\_i$ of $B\_s$ and present your results in the following table:

| | |
|---|---|
| $L(13, 15)$ | |
| $L\_i(13, 15)$ | |
| Memory used for $L$ (in bytes) | |
| Memory used for $L\_i$ (in bytes) | |

Use the following options to compute the incomplete Cholesky factorization $L\_i$:

```
options = struct('type','ict','droptol',1e-02, ...
                 'michol','off')
```

(c) Use the options

```
tol = 10^(-12); maxit = 2;
```

in Matalab's `pcg` command without any preconditioner to compute an approximation to $\mathbf{x}$ for $B\mathbf{x} = \mathbf{b}$. Present your results in the following table:

| | |
|---|---|
| Value of $x_8$ | |
| flag | |
| relres | |
| iter | |

Then use the outputs from `pcg` to explain whether the approximate vector generated by `pcg` here is a good approximation to the solution to $B\mathbf{x} = \mathbf{b}$.

(d) Use the options

```
tol = 10^(-12); maxit = 100;
```

in Matalab's `pcg` command without any preconditioner to compute an approximation to $\mathbf{x}$ for $B\mathbf{x} = \mathbf{b}$. Present your results in the following table:

| | |
|---|---|
| Value of $x_9$ | |
| flag | |
| relres | |
| iter | |

Then use the outputs from `pcg` to explain whether the approximate vector generated by `pcg` here is a good approximation to the solution to $B\mathbf{x} = \mathbf{b}$.

(e) Use the options

```
tol = 10^(-12); maxit = 100;
```

and the incomplete Cholesky factorization $L\_i$ as the preconditioner in Matalab's `pcg` command to compute an approximation to $\mathbf{x}$ for $B\mathbf{x} = \mathbf{b}$. Present your results in the following table:

| | |
|---|---|
| Value of $x_{10}$ | |
| flag | |
| relres | |
| iter | |

Then use the outputs from `pcg` to explain whether the approximate vector generated by `pcg` here is a good approximation to the solution to $B\mathbf{x} = \mathbf{b}$ and whether using a precondition is advantageous.

13. Download the data files `I_512_1.mat`, `J_512_1.mat`, `V_512_n.mat`, and `b_512_n.mat` from our course web page on Canvas and load them into Matlab. This will generate 4 arrays `I, J, V, b` in Matlab. Let $A$ be the sparse matrix generated by the arrays $I, J$ and $V$.

    (a) Solve the linear system $A\mathbf{x} = \mathbf{b}$ by the GMRES method with a preconditioner generated by incomplete LU factorization.
    Use the following parameters for generating the preconditioner:

    ```
    struct('type','nofill','droptol',1e-7)
    ```

    Use the following parameters for the GMRES method:

    ```
    restart = 100; tol = 10e-12; maxit = 5000;
    ```

    Present your numerical result in the following table:

| | |
|---|---|
| Value of $x_{8521}$ | |
| flag | |
| relres | |
| iter | |

    (b) Solve the linear system $A\mathbf{x} = \mathbf{b}$ by the GMRES method without a preconditioner.
    Use the following parameters for the GMRES method:

    ```
    restart = 100; tol = 10e-12; maxit = 5000;
    ```

    Present your numerical result in the following table:

| | |
|---|---|
| Value of $x_{8521}$ | |
| flag | |
| relres | |
| iter | |

(c) Use your experiences in the two computations above to comment on the GMRES method.

14. Consider the function $f(x)$ given in Problem 2.3 on page 42 of the textbook.

(a) Implement a user-defined Matlab function for $f(x)$ that takes

$$\beta, a_1, a_2, a_3, a_4$$

as parameter inputs in addition to the main variable input $x$.

(b) For $\beta = 0$, $a_1 = 10, a_2 = 13, a_3 = 8, a_4 = 10$, make a plot of $f(x)$ for $x \in [0, \pi]$. Then, choose a suitable starting interval $[a, b]$ to find $\alpha \in [0, \pi]$ such that $f(\alpha) = 0$ by the bisection method with `tol = 10^(-12)`. Present the computational results in the following table:

| | |
|---|---|
| $a$ | |
| $b$ | |
| $\alpha$ | |
| residual | |
| iterations used | |

Also, present your Maltab script used to generate data in this table. Make sure that your script uses the "varargin" functionality of "bisection.m".

(c) For $\beta = 0$, $a_1 = 9, a_2 = 12, a_3 = 9, a_4 = 9$, make a plot of $f(x)$ for $x \in [0, \pi]$. Then, choose a suitable starting interval $[a, b]$ to find $\alpha \in [0, \pi]$ such that $f(\alpha) = 0$ by the bisection method with `tol = 10^(-12)`. Present the computational results in the following table:

| | |
|---|---|
| $a$ | |
| $b$ | |
| $\alpha$ | |
| residual | |
| iterations used | |

Also, present your Maltab script used to generate data in this table. Make sure that your script uses the "varargin" functionality of "bisection.m".

15. Consider the function $f(x)$ given in Problem 2.3 on page 42 of the textbook.

    (a) Implement user-defined Matlab functions for $f(x)$ and $f'(x)$, respectively, that take $\beta, a_1, a_2, a_3, a_4$ as parameter inputs in addition to the main variable input $x$.

    (b) For $\beta = 1.5$, $a_1 = 10, a_2 = 13, a_3 = 8, a_4 = 10$, make a plot of $f(x)$ for $x \in [0, \pi]$. Then, choose a suitable starting point $x^{(0)}$ to find $\alpha$ such that $1 \leq \alpha \leq \pi$ and $f(\alpha) = 0$ by the Newton method with `tol = 10^(-12)`. Present the computational results in the following table:

| | |
|---|---|
| $x^{(0)}$ | |
| $\alpha$ | |
| residual | |
| iterations used | |

    Also, present your Maltab script used to generate data in this table. Make sure that your script uses the "varargin" functionality of "newton.m".

    (c) For $\beta = 1.5$, $a_1 = 10, a_2 = 13, a_3 = 8, a_4 = 10$, make a plot of $f(x)$ for $x \in [0, \pi]$. Then, choose a suitable starting point $x^{(0)}$ to find $\alpha$ such that $0 \leq \alpha \leq 1$ and $f(\alpha) = 0$ by the Newton method with `tol = 10^(-12)`. Present the computational results in the following table:

| $x^{(0)}$ | |
|---|---|
| $\alpha$ | |
| residual | |
| iterations used | |

Also, present your Maltab script used to generate data in this table. Make sure that your script uses the "varargin" functionality of "newton.m".

16. Implement the secant method in a Matlab function whose interface is as follows:

```
function [zero,res,niter]=secant(fun, x0, x1, ...
                        tol, nmax, varargin)
```

Test your program by using it to compute a zero for

$$f = x^3 + 4x^2 - 10.$$

Use $tol = 10^{-12}$, $nmax = 10000$, and the values for $x^{(0)}$ and $x^{(1)}$ in the example given in lecture slides. Present computational results in a table as follows:

| Zero found | Residual | Actual iterations used |
|---|---|---|
| | | |

17. Apply the secant method to compute the negative zero with the error tolerance $tol = 1.0e - 12$ for

$$f(x) = cosh(x) + cos(x) - 3.$$

Explain how the initial guesses $x_0$ and $x_1$ are chosen and present computational results in a table as follows:

| $x^{(0)}, x^{(1)}$ | |
|---|---|
| Zero found | |
| Residual | |
| Iterations used | |

18. Consider the following system of nonlinear equations about $x_1$ and $x_2$:

$$\begin{aligned} sin(a_1 x_1 x_2) - 2x_2 - x_1 &= 0, \\ a_2(e^{2x_1} - e) + a_3(4x_2^2 - 2x_1) &= 0. \end{aligned}$$

(a) Put this system of nonlinear equations in vector form $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ where $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}))^T$ and identify the formulas for $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$.

(b) Derive the Jacobian for $\mathbf{f}(\mathbf{x})$.

(c) Implement user-defined Matlab functions for $\mathbf{f}(\mathbf{x})$ and its Jacobian, respectively, that take $a_1, a_2, a_3$ as parameter inputs in addition to the main variable input $\mathbf{x}$.

(d) Compute $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ in Newton's method for this nonlinear system with the following parameters:

$$a_1 = 4\pi, \quad a_2 = \left(\frac{4\pi - 1}{4\pi}\right), \quad a_3 = e = exp(1).$$

Start the iteration with $\mathbf{x}^{(0)} = [-0.3; 0.05]^T$. Present numerical results in the following table:

| $k$ | $x_1^{(k)}$ | $x_2^{(k)}$ |
|---|---|---|
| 1 | | |
| 2 | | |

(e) Use Newton's method with $tol = 10^{-8}$ and $\mathbf{x}^{(0)} = [-0.3; 0.05]^T$ to solve the nonlinear system described in (d) above and present numerical results in a table as follows:

| $x_1$ | $x_2$ | Residual | Actual iterations |
|---|---|---|---|
| | | | |

Also, present your Maltab script used to generate data in this table. Make sure that your script uses the varargin functionality of newtonsys.m.

19. This problem is for implementing and testing the modified Newton's method with the approximate Jacobian constructed by the finite difference method with two approximate points.

(a) Use the Matlab function `approxJ_fd.m` given in the Lecture Slides to compute an approximate Jacobian $\tilde{J}_{\mathbf{f}}$ for the system of nonlinear equations in Problem 18 at $\mathbf{x}^{(1)} = [-0.3, 0.05]^T$ with the following parameters:

$$\mathbf{x}^{(0)} = [-0.31, 0.051]^T, \quad a_1 = 12.56, \quad a_2 = 0.92, \quad a_3 = 2.72.$$

Present numerical results in the following table

| $\tilde{J}(1,1)$ | $\tilde{J}(2,2)$ |
|---|---|
| | |

(b) Compute $\mathbf{x}^{(2)}$ in the modified Newton's method with the approximate Jacobian constructed by two approximate points for this system of nonlinear equations. Use $\mathbf{x}^{(0)}, \mathbf{x}^{(1)}$ and parameters specified in 19a and present the numerical results in the following table

| $x_1^{(2)}$ | $x_2^{(2)}$ |
|---|---|
| | |

20. This problem is for implementing and testing the modified Newton's method with the approximate Jacobian constructed by the finite difference method with one approximate point.

(a) Use the Matlab function `approxJ_fdh.m` given in the Lecture Slides to compute an approximate Jacobian $\tilde{J}_{\mathbf{f}}$ for the system of nonlinear equations in Problem 18 at $\mathbf{x}^{(0)} = [-0.3, 0.05]^T$ with the following parameter:

$$h = 0.001, \quad a_1 = 12.56, \quad a_2 = 0.92, \quad a_3 = 2.72.$$

Present numerical results in the following table

| $\tilde{J}(1,2)$ | $\tilde{J}(2,1)$ |
|---|---|
| | |

(b) Implement the modified Newton's method that uses `approxJ_fdh.m` to generate the approximate Jacobian. Make sure that the implemented Matlab function has the following interface:

```
function [x, res, niter] = newtonsys_approxJ_fdh(Ffun, ...
                          x0, h, tol, nmax, varargin)
```

(c) Use the Matlab function newtonsys_approxJ_fdh.m implemented above to solve the system of nonlinear equations specified in 20a with

```
tol = 10^(-8); nmax = 100;
```

Present the numerical results in the following table

| | |
|---|---|
| $x_1$ | |
| $x_2$ | |
| res | |
| niter | |

21. This problem is for implementing and testing the modified Newton's method with the approximate Jacobian constructed by the complex variable method.

(a) Implement the complex variable method for computing an approximate Jacobian $\tilde{J}_{\mathbf{f}}$ at a point $\mathbf{x}$. Make sure that the implemented Matlab function has the following interface:

```
function J = approxJ_compl(fun, x, EPS, varargin)
```

(b) Use the Matlab function approxJ_compl.m implemented above to compute an approximate Jacobian $\tilde{J}_{\mathbf{f}}$ for the system of nonlinear equations in Problem 18 at $\mathbf{x}^{(0)} = [-0.3, 0.05]^T$ with the following parameter:

$$EPS = 10^{-1}, \quad a_1 = 12.566, \quad a_2 = 0.920, \quad a_3 = 2.718.$$

Present numerical results in the following table

| $\tilde{J}(1,2)$ | $\tilde{J}(2,1)$ |
|---|---|
| | |

(c) Implement the modified Newton's method that uses approxJ_compl.m to generate the approximate Jacobian. Make sure that the implemented Matlab function has the following interface:

```
function [x, res, niter] = newtonsys_approxJ_compl(Ffun, ...
                          x0, EPS, tol, nmax, varargin)
```

(d) Use the Matlab function newtonsys_approxJ_compl.m implemented above to solve the system of nonlinear equations specified in 21b with

```
EPS = 10^(-5); tol = 10^(-8); nmax = 100;
```

Present the numerical results in the following table

| | |
|---|---|
| $x_1$ | |
| $x_2$ | |
| res | |
| niter | |

22. This problem is for implementing and testing the Broyden's method.

(a) Implement the Broyden's method described in the Lecture Slides. Make sure that the implemented Matlab function has the following interface:

```
function [x, res, niter, err, B]=broyden(Ffun, B0, ...
                    x0, tol, nmax, varargin)
```

(b) Use the Broyden's method implement above and suitable choices of `nmax` to compute $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ for the system of nonlinear equations in Problem 18 with the following parameters:

$$a_1 = 12.5, \quad a_2 = 0.9, \quad a_3 = 2.7.$$

Use

$$\mathbf{x}^{(0)} = [-0.3, 0.05]^T, \quad B_0 = I_{2\times 2}, \quad tol = 10^{-8}$$

Present numerical results in the following table

| $k$ | $x_1^{(k)}$ | $x_2^{(k)}$ |
|---|---|---|
| 1 | | |
| 2 | | |

(c) Use the Broyden's method to solve the system of nonlinear equations specified in 22b. Use $B_0$ as the approximate Jacobian at $\mathbf{x}^{(0)} = [-0.3, 0.05]^T$ generated by the complex variable method with $EPS = 10^{-7}$ and use

```
tol = 10^{-8}; nmax = 100;
```

Present the numerical results in the following table

| | |
|---|---|
| $x_1$ | |
| $x_2$ | |
| res | |
| err | |
| niter | |

23. Exercise 3.3 from the textbook. Present your Matlab script for generating the numerical results and put numerical results in a table as follows:

| year | WE life expectation | EE life expectation |
|---|---|---|
| 1977 | | |
| 1983 | | |
| 1988 | | |

24. Implement the Matlab function for the $k$-th Lagrange characteristics polynomial $\phi_{n,k}(x)$ or its first derivative $\phi'_{n,k}(x)$ such that its interface is as follows:

```
function f = lagrange_char(x, x_nodes, k, d_order)
```

where `x_nodes` is the vector representing the given $n+1$ nodes:

$$x_0 < x_1 < x_2 < \cdots < x_n$$

Then, apply your code to the following nodes:

| $x$ | -55 | -25 | 5 | 35 | 65 |
|---|---|---|---|---|---|

Present your numerical results in a table as follows:

| | |
|---|---|
| $\phi_2(7)$ | |
| $\phi_2'(7)$ | |
| $\phi_5(7)$ | |
| $\phi_5'(7)$ | |

Implement the program for the Lagrange interpolating polynomial such that it have the following interface:

```
function p = lagrange_interp(z, x, y, d_order)
```

where `z` is variable of the interpolating polynomial, `x` and `y` provide the data for interpolation, and `d_order` is the derivative of the interpolating polynomial. Test your program against the numerical results given in lecture slides with the following data:

| $x$ | -55 | -25 | 5 | 35 | 65 |
|---|---|---|---|---|---|
| $y$ | -3.25 | -3.2 | -3.02 | -3.32 | -3.1 |

Present your test results in a table as follows:

| $z$ | $\Pi_4(z)$ | $\Pi_4'(z)$ |
|---|---|---|
| 20.5 | | |

25. In this problem we will use the temperature data (given in Table 3.1) at latitudes

$$-55, -25, -5, 25, 55$$

for the carbon acid concentration $K = 1.5$ to find an approximation to the temperature distribution.

(a) Use the interpolating polynomial with the data specified above to make a plot for visualizing the temperature for latitudes between $-65$ and 65. Scale your plot by the following Matlab command:

```
axis([-60, 70, 2.2, 4])
```

(b) Use this interpolating polynomial to find approximations to the temperature at latitudes $-65, -17, 0, 17, 65$ and present your data in a table as follows:

| Latitudes | Temperature |
|---|---|
| $-65$ | |
| $-17$ | |
| 0 | |
| 17 | |
| 65 | |

Are these approximate temperatures reliable? Why? Some of them are more reliable than the other?

26. Let $P_4(x)$ be the interpolating polynomial constructed with the temperature data (given in Table 3.1) at latitudes

$$-55, -25, -5, 25, 55$$

for the carbon acid concentration $K = 1.5$.

(a) Assume we used Newton's method to solve $P_4(x) - 3.225 = 0$ starting from $x^{(0)} = -25$. Compute $x^{(2)}$ in Newton's iteration and present numerical results in the following table:

| $x^{(2)}$ | $P_4(x^{(2)})$ | $P_4'(x^{(2)})$ |
|---|---|---|
| | | |

(b) Find the latitude where the temperature is 3.225 by solving $P_4(x) - 3.225 = 0$ with Newton's method. Present numerical results in the following table:

| Latitude | $x^{(0)}$ used | Num. of iterations |
|---|---|---|
| | | |

Use more rows in the table above to present results if more latitudes are found.

27. (a) Use the interpolating polynomial $P(x)$ constructed with all the data related to the value $K = 1.5$ given in Table 3.1 in the textbook to make a plot for visualizing the temperature over the given latitude range. Present your Matlab script for making this plot and scale your plot by

```
axis([-60, 70, 2.2, 4])
```

(b) Use the interpolating polynomial above to predict temperatures at the following latitudes:

$$-50, -20, -10, 0, 10, 17$$

Present numerical results in a table as follows:

| Latitude | Temperature |
|---|---|
| $-50$ | |
| $-20$ | |
| $-10$ | |
| $-0$ | |
| 10 | |
| 17 | |

(c) Use the cubic spline interpolation constructed $S(x)$ with all the data related to the value $K = 1.5$ given in Table 3.1 in the textbook to make a plot for visualizing the temperature over the given latitude range. Use the following boundary conditions for this cubic spline interpolation:

$$S'(-55) = 0.003979604851107, \quad S'(65) = -0.010471912543415.$$

Present your Matlab script for making this plot and scale your plot by the following Matlab command:

```
axis([-60, 70, 2.2, 4])
```

(d) Use the cubic spline interpolation above to predict temperatures at following the latitudes:

$$-50, -20, -10, 0, 10, 17.$$

Present numerical results in a table as follows:

| Latitude | Temperature |
|---|---|
| $-50$ | |
| $-20$ | |
| $-10$ | |
| $-0$ | |
| 10 | |
| 17 | |

(e) Use your work above to discuss the interpolating polynomial and cubic spline interpolation techniques.

28. Let $S(x)$ be the cubic spline interpolation with all the data related to the value $K = 1.5$ given in Table 3.1 in the textbook and the the following boundary conditions:

$$S'(-55) = 0.003979604851107, \quad S'(65) = -0.010471912543415.$$

(a) Assume we use Newton's method to be started from $x^{(0)} = 5$ to find the latitude at where the temperature is 3.175 with the above cubic spline $S(x)$ to model the temperature as a function of the latitude $x$. Compute $x^{(2)}$ in Newton's iteration and present numerical results in the following table:

| $x^{(2)}$ | $S(x^{(2)})$ | $S'(x^{(2)})$ |
|---|---|---|
| | | |

(b) Use Newton's method to find the latitude where the temperature is 3.175 with the above cubic spline $S(x)$ to model the temperature. Present numerical results in the following table:

| Latitude | $x^{(0)}$ used | Num. of iterations |
|---|---|---|
| | | |

Use more rows in the table above to present results if more latitudes are found.

29. (a) Use the natural cubic spline function for the data listed in Table 3.2 in the textbook to find approximations to $\epsilon(0.15), \epsilon(0.35), \epsilon(0.65)$

and $\epsilon(0.8)$. Present the Matlab script for generating these approximations and present numerical results in a table as follows:

| $\sigma$ | $\epsilon(\sigma)$ |
|---|---|
| 0.15 | |
| 0.35 | |
| 0.65 | |
| 0.80 | |

(b) Use the natural cubic spline function for the data listed in Table 3.2 in the textbook to make a plot for the function $\epsilon(\sigma)$ for $0 \leq \sigma \leq 0.7$. Scale your plot by

```
axis([0, 0.7, 0, 0.31])
```

30. (a) Implement the Lagrange formula for the Hermite interpolating polynomial $H_{2n+1}(z)$ such that it has the following interface:

```
function H = hermite_interp(z, x, y, ty, d_order)
```

(b) Let $f(x) = sin(exp(x))$ and consider the following data table:

| $x$ | 0 | 1/2 | 1 |
|---|---|---|---|
| $f(x)$ | $f(0)$ | $f(1/2)$ | $f(1)$ |
| $f'(x)$ | $f'(0)$ | $f'(1/2)$ | $f'(1)$ |

Let $H_5(z)$ be the Hermite interpolating polynomial of this data table. Use the Matlab function `hermite_interp` to compute $H_5(1/4)$, $H_5'(1/4)$, $H_5(3/4)$ and $H_5'(3/4)$. Present numerical results in a table as follows:

| $x$ | $H_5(x)$ | $H_5'(x)$ |
|---|---|---|
| 1/4 | | |
| 3/4 | | |

(c) Let $H(z)$ be the piecewise cubic Hermite interpolating polynomial of the data table above. Use the Matlab function `hermite_interp` to compute $H(1/4), H'(1/4), H(3/4)$ and $H'(3/4)$. Present numerical results in a table as follows:

| $x$ | $H(x)$ | $H'(x)$ |
|---|---|---|
| 1/4 | | |
| 3/4 | | |

31. Use Mathematica to identify the order of accuracy with respect to $h$ for those formulas in Exercise 4.3b of the textbook. Note that $x_i$s in this formula are equispaced such that $x_i = x, x_{i-1} = x - h, x_{i-2} = x - 2h, x_{i+1} = x + h$.

32. Consider the data table given in Exercise 4.4 of the textbook (on page 134) about $n(t)$ for the number of individuals of a given population. Compute $n'(1.5)$ and its approximations by finite difference formulas specified in the table below and use the numerical result to fill in the table:

| | |
|---|---|
| $n'(1.5)$ | |
| $D_f^1(n, 1.5, 0.5)$ | |
| $D_b^1(n, 1.5, 0.5)$ | |
| $D_c^1(n, 1.5, 0.5)$ | |
| $D_{4.3a}f(1.5)$ | |
| $D_{4.3b}f(1.5)$ | |

where $D_{4.3a}f(x)$ and $D_{4.3b}f(x)$ are the finite difference formulas for approximating $f'(x)$ given in Exercise 4.3 of the textbook (page 134).

33. (a) Implement Matlab functions for generating the nodes and weights of the closed Newton-Cotes formula. These Matlab function should have the following interfaces:

```
function x = newton_cotes_closed_nodes(a, b, n)
function w = newton_cotes_closed_weights(a, b, n)
```

and they can handle $n = 1, 2, 3, \cdots, 5$.

(b) Implement the close Newton-Cotes formula in a Matla function such that its interface is as follows:

```
function int_val = quadrature_NCC(f_name, a, b, n, varargin)
```

(c) Use the closed Newton-Cotes formula with $n = 5$ to compute the following integrals

$$\int_0^{0.5} sin(\beta x^2)dx, \quad \beta = \sqrt{2}, \pi.$$

Present numerical results in a table as follows:

| $\int_0^{0.5} sin(\sqrt{2}\, x^2)dx$ | $\int_0^{0.5} sin(\pi\, x^2)dx$ |
|---|---|
| | |

Present the Matlab script for the related computations. Make sure that the `varargin` functionality of the Matlab program `quadrature_NCC.m` is used in the computations.

34. (a) Implement Matlab functions for generating the nodes and weights of the Gauss-Legendre formula on the reference interval $[-1, 1]$. These Matlab function should have the following interfaces:

```
function x = gaussian_quadr_nodes_ref(n)
function w = gaussian_quadr_weights_ref(n)
```

and they can handle $n = 1, 2, 3, \cdots, 5$.

(b) Use the Gauss-Legendre formula with $n = 4$ to compute the following integrals

$$\int_0^{0.5} cos(\beta x^2)dx, \quad \beta = \sqrt{2}, \pi.$$

Present numerical results in a table as follows:

| $\int_0^{0.5} cos(\sqrt{2}\, x^2)dx$ | $\int_0^{0.5} cos(\pi\, x^2)dx$ |
|---|---|
| | |

Present the Matlab script for the related computations. Make sure that the `varargin` functionality of the Matlab program `quadrature_GL.m` is used in the computations.

35. Use the composite Newton-Cotes formula with 4 nodes on each subinterval to compute following integral:

$$\int_0^2 cos(\pi x^2) sin(\sqrt{2}\ x^2)dx.$$

Present numerical results in the following table:

| Number of subintervals | $\int_0^2 cos(\pi x^2) sin(\sqrt{2}\ x^2) dx$ |
|---|---|
| 20 | |
| 40 | |
| 80 | |

36. Use the composite Gauss-Legendre formula with 3 nodes on each subintervals to compute following integral:

$$\int_0^2 cos(\sqrt{2}x^2) sin(\pi\ x^2) dx.$$

Present numerical results in the following table:

| Number of subintervals | $\int_0^2 cos(\sqrt{2}x^2) sin(\pi\ x^2) dx$ |
|---|---|
| 20 | |
| 40 | |
| 80 | |

37. Use the composite Gauss-Legender formula with 50 subintervals and 3 local nodes to compute the area between the curves of

$$g_1(x) = cos(x^2), \quad g_2(x) = exp(-sin(x^2)) + cos(x)$$

over the interval $[0, \pi]$.

38. Use the composite Gauss-Legender formula with 50 subintervals and 3 local nodes to compute the following integral

$$\iint_R exp\left(-sin(x^2/10) - y^2/2\right) dA$$

where $R$ is the region formed between the curves described by the following functions:

$$g_1(x) = cos(x^2), \quad g_2(x) = exp(-sin(x^2)) + cos(x), \quad x \in [0, \pi].$$

39. Use the composite Gauss-Lengedre formula with 3 local nodes to compute the area of the surface on the hemisphere described by $x^2+y^2+z^2 = 9, \quad z \geq 0$ that lies above the region

$$R = \{(x, y) \mid 0 \leq x \leq 1, 0 \leq y \leq 1\}.$$

Recall that the area of the surface described by $z = f(x, y)$ for $(x, y)$ in a region $R$ is given by

$$\iint_R \sqrt{\big(f_x(x,y)\big)^2 + \big(f_y(x,y)\big)^2 + 1}\ dA$$

40. Consider the following parametric curve:

$$\begin{cases} x(t) = acos(t)(b + cos(t)), \\ y(t) = asin(t)(c + cos(t)), \end{cases} \qquad t \in [0, 2\pi].$$

with $a = 1, b = 0.5, c = 0.5$.

(a) Use Matlab to make a plot of this curve. Scale the plot by

```
axis([-0.5, 2.5, -1.5, 1.5])
axis equal
```

(b) Compute the arch length for this curve. Recall that the arch length of a parametric curve $(x(t), y(t)), \quad t = [t_s, t_e]$ is given by

$$L = \int_{t_s}^{t_e} \sqrt{\big(x'(t)\big)^2 + \big(y'(t)\big)^2}\ dt.$$

41. (a) (5 points) Implement the classic 4-stage Runge-Kutta method by modifying the Matlab function `ode_rk33.m`. The interfaced of the Matlab function for this 4 stage Runge-Kutta method should be

```
[t, u] = ode_rk4_classic(odefun, tspan, y0, Nh, varargin);
```

(b) (10 points) Consider the following IVP:

$$y'(t) = asin(y(t)) + be^{-t/2},\ t \in (0, 1],$$
$$y(0) = 1.$$

For $a = 1, b = 1$, solve this IVP with the methods specified in the following table by partitioning the time interval into 25 subintervals. The function $f(t, y)$ for this problem should be implemented to take parameters $a$ and $b$ and these parameters should be passed to ODE solvers via `varargin`. Present numerical solutions by filling the table below:

| $t$ | $y(t)$ by 3-stage RK method | $y(t)$ by 4-stage RK method |
|---|---|---|
| 0.28 | | |
| 0.76 | | |
| 1.00 | | |

Present your Matlab script for generating these numerical results.

(c) (10 points) Then, for $a = 2, b = 5$, resolve this IVP by the methods specified in the following table by partitioning the time interval into 25 subintervals. Present numerical solutions with the data required in the table.

| $t$ | $y(t)$ by 3-stage RK method | $y(t)$ by 4-stage RK method |
|---|---|---|
| 0.28 | | |
| 0.76 | | |
| 1.00 | | |

Present your Matlab script for generating these numerical results.

(d) (5 points) Use the numerical results in Problem 41c and the given ODE to compute $y'(0)$ and $y'(1)$ and present your result in the following table:

| | |
|---|---|
| $y'(0)$ by RK3 | |
| $y'(0)$ by RK4 | |
| $y'(1)$ by RK3 | |
| $y'(1)$ by RK4 | |

(e) (10 points) Use the numerical solutions generated in Problem 41c (not just those numbers presented in the table there) and the cubic spline interpolation with the clamped boundary condition to find approximations to $y(t)$ at $t$ specified in the following table:

| $t$ | $y(t)$ by $S_3(t)$ | $y(t)$ by $S_4(t)$ |
|---|---|---|
| 0.55 | | |
| 0.91 | | |

where $S_3(t)$ and $S_4(t)$ are the cubic spline interpolations of the numerical solutions generated by the 3-stage RK method and the classic 4-stage RK method with the clamped boundary condition, respectively.

(f) (10 points) Let $A$ be the area between the curve of $y(t)$ and $g(t) = 1$ over the interval $0 \leq t \leq 1$. Use the cubic spline interpolations describe in Problem 41e to compute $A$ by the composite Gauss-Lengendre method with 25 subintervals and 3 local nodes. Present numerical results in the following table:

| $A$ by $S_3(t)$ | $A$ by $S_4(t)$ |
|---|---|
| | |

(g) (10 points) Let $L$ be the arc length of the curve of the function $y(t)$ over the interval $0 \leq t \leq 1$. Use the cubic spline interpolations describe in Problem 41e to compute $L$ by the composite Gauss-Lengendre method with 25 subintervals and 3 local nodes. Present numerical results in the following table:

| $L$ by $S_3(t)$ | $L$ by $S_4(t)$ |
|---|---|
| | |

42. In a mechanical system, the locations of two objects are modeled by the following IVP:

$$\begin{aligned}
&(m_1 + 0.5m_2)x_1''(t) - 0.5m_2x_2''(t) + k_1x_1(t) = 0,\\
&-0.5m_2x_1''(t) + 1.5m_2x_2''(t) + k_2x_2(t) = 0,\\
&x_1(0) = 1.5, x_2(0) = 0, x_1'(0) = 0, x_2'(0) = 0,
\end{aligned}$$

for $t \in (0, 10]$.

(a) (5 points) Rewrite the given system of 2nd order ODEs as

$$\begin{aligned}
x_1''(t) &= f_1(t, x_1(t), x_1'(t), x_2(t), x_2'(t)),\\
x_2''(t) &= f_2(t, x_1(t), x_1'(t), x_2(t), x_2'(t)).
\end{aligned}$$

(b) (5 points) Convert the given IVP to an IVP for system of first order ODEs.

(c) (10 points) Assume

$$m_1 = m_2 = 5, k_1 = k_2 = 15.$$

Solve the given IVP by the classic 4 stage Runge-Kutta method with 1000 subintervals in $0 \leq t \leq 10$ and present your numerical solution with the following table:

| $t$ | 2 | 8 |
|---|---|---|
| $x_1(t)$ | | |
| $x_1'(t)$ | | |
| $x_2(t)$ | | |
| $x_2'(t)$ | | |

(d) (5 points) Make a plot for the curves of $x_1(t)$ and $x_2(t)$. Scale the plot by

```
axis([0, 10, -1.6, 1.6])
```

(e) (5 points) Make a plot for the curves of $x_1'(t)$ and $x_2'(t)$. Scale the plot by

```
axis([0, 10, -2.5, 2.5])
```

(f) (5 points) Make a plot for the phase plane curve for $x_1(t)$ and $x_2(t)$. Scale the plot by

```
axis([-1.6, 1.6, -1.6, 1.6])
```

(g) (10 points) Use the numerical solution generated in Problem 42c to find approximations to $x_1''(t)$ and $x_2''(t)$ at $t$ specified in the following table:

| $t$ | $x_1''(t)$ | $x_2''(t)$ |
|---|---|---|
| 0 | | |
| 2 | | |
| 10 | | |

(h) (10 points) Use the cubic spline interpolation and the composite integration formula with 1000 subintervals and 3 local nodes to compute the arc length for the phase plane curve generated in Problem 42f.