CS6210 - Homework/Assignment-5

Arnab Das(u1014840)

November 28, 2016

1: Chapter-9: Question-5

An $n \times n$ linear system of equations $Ax = b$ is modified such that every b_i is replaced with $b_i - x_i^3$.
(a) For the system $Ax = b$, thus we have:

A = $\begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{bmatrix}$, $x = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix}$ and $b = \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_n \end{bmatrix}$

The b vector gets modified as $\begin{bmatrix} b_1 - x_1^3 \\ b_2 - x_2^3 \\ \dots \\ b_n - x_n^3 \end{bmatrix}$, so that now we have a non-linear system that can be expressed as **f(x)=**, where f is a vector of functions represented as :

$$f_1(x) = (a_{11}x_1 + x_1^3) + a_{12}x_2 + \dots + a_{1n}x_n - b_1 = 0$$
$$f_2(x) = a_{21}x_1 + (a_{22}x_2 + x_2^3) + \dots + a_{2n}x_n - b_2 = 0$$
$$\dots$$
$$f_i(x) = a_{i1}x_1 + a_{i2}x_2 + \dots + (a_{ii}x_i + x_i^3) + \dots + a_{in}x_n - b_i = 0$$
$$\dots$$
$$f_n(x) = a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n + x_n^3 - b_n = 0$$

Then, the Jacobian of this vector of functions, f, will be:

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} & \dots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3} & \dots & \frac{\partial f_2}{\partial x_n} \\ \dots & \dots & \dots & \dots & \dots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \frac{\partial f_n}{\partial x_3} & \dots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$

$$= \begin{bmatrix} (a_{11} + 3x_1^2) & a_{12} & a_{13} & \dots & a_{1i} & \dots & a_{1n} \\ a_{21} & (a_{22} + 3x_2^2) & a_{23} & \dots & a_{2i} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a_{i1} & a_{i2} & a_{i3} & \dots & (a_{ii} + 3x_i^2) & \dots & a_{in} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{ii} & \dots & (a_{in} + 3x_n^2) \end{bmatrix} = \text{A + D, where D is:}$$

$$D = \begin{bmatrix} 3x_1^2 & 0 & 0 & \dots & 0 & \dots & 0 \\ 0 & 3x_2^2 & 0 & \dots & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 3x_i^2 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 0 & \dots & 3x_n^2 \end{bmatrix}$$

(b) Given that A is strictly diagonally dominant with positive elements in its diagonal. We know that a strictly diagonally dominant matrix is non-singular, hence **A is non-singular**. Now, diagonal dominance means:

$$\forall i, |a_{ii}| \geq \sum_{j=1}^{n} |a_{ij}|, j \neq i$$

Since, D is a diagonal matrix with non-negative elements along its diagonal, hence adding D to A, conserves the diagonal dominance in the resulting matrix J. Thus, at any iterate, the Jacobian, J, also remains diagonally dominant and hence non-singular.

(c) In Newton's method, we are theoretically basically solving the following:

$$x_{k+1} - x_k = -J(x_k)f(x_k) \tag{1}$$

where, x_k is the value of x in the k'th iteration. For convergence of the method, we want the following:

$$\frac{||x_{k+1} - x^*||}{||x_k - x^*||} = \lambda \tag{2}$$

where, λ is some positive constant, and x^* is the optimal solution.
The error at the (k+1)'th iteration can be written as:

$$||e_{k+1}|| = ||x_{k+1} - x^*|| = ||x_k - J(x_k)^{-1}f(x_k) - x^*|| = ||e_k - J(x_k)^{-1}f(x_k)|| \tag{3}$$

Now, from Taylor's series expansion, we have:

$$f(x_k) = f(x^*) + J(x_k)(x_k - x^*) + \frac{1}{2}(x_k - x^*)^T H(x_k)(x_k - x^*) + \dots$$

Approximating higher order terms:

$$f(x_k) = f(x^*) + J(x_k)e_k + \frac{1}{2}(e_k)^T H(x_k)(e_k)$$

At the optima, $f(x^*) = 0$. Multiplying both sides by $J(x_k)^{-1}$ thus we get:

$$J(x_k)^{-1}f(x_k) = 0 + e_k + \frac{1}{2}J(x_k)^{-1}(e_k)^T H(x_k)(e_k)$$

Replacing this in (3)

$$||e_{k+1}|| = ||e_k - e_k - \frac{1}{2}J(x_k)^{-1}(e_k)^T H(x_k)(e_k)|| = ||\frac{1}{2}J(x_k)^{-1}(e_k)^T H(x_k)(e_k)|| \le \frac{||J^{-1}||||H||}{2}||e_k||^2 \tag{4}$$

Thus (4) indicates that if the Jacobian is non-singular, then it quadratically converges. So, the only remaining peice is to show that the Jacobian is non-singular. We know that a symmetric positive definite matrix is non-singular. If we can proove that J is symmetric positive definite, then it suites our purpose: Consider any non-zero vector, **u**, then:

$$u^T Ju = u^T(A + D)u = u^T Au + u^T Du$$

we know that A is symmetric positive definite, hence $u^T Au > 0$. So, we need to show now $u^T Du > 0$.

$$u^T Du = \begin{bmatrix} u_1 & u_2 & \dots & u_n \end{bmatrix} \begin{bmatrix} 3x_1^2 & 0 & 0 & \dots & 0 \\ 0 & 3x_2^2 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 3x_n^2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \dots \\ u_n \end{bmatrix} = x_1^2u_1^2 + x_2^2u_2^2 + \dots + x_i^2u_i^2 + \dots + x_n^2u_n^2 > 0$$

Hence,

$$u^T Ju = u^T(A + D)u = u^T Au + u^T Du > 0$$

Thus, the Jacobian is also symmetric positive definite and hence invertible and non-singular. Hence, it is guaranteed to converge as per (4).

2: Chapter-9: Question-6

(a) Given a linear system of equations, $Ax = b$, on which we apply Newton's method. For a linear system, the Jacobian, J, will be : J = $\begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \frac{\partial f_n}{\partial x_3} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$

$= \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1i} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2i} & \cdots & a_{2n} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{i1} & a_{i2} & a_{i3} & \cdots & a_{ii} & \cdots & a_{in} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{ii} & \cdots & a_{in} \end{bmatrix} = \text{A}$

Thus, Newton's method applied to this problem becomes:

$$x^* = x_1 = x_0 + A^{-1}(b - Ax_0) = A^{-1}b$$

Hence, it will converge in a single iteration.

(b) The Newton's method is of the form:

$$J(x_k)(x_{k+1} - x_k) = -f(x_k)$$

If the Jacobian is singular at the solution, then theoretically it means $\Delta x_k = x_{k+1} - x_k$, will be potentially large in the region close to the solution which means the values of x at the k'th and (k+1)'th iterate will have a large difference implying that convergence will be extremely slow and even the convergence might be lost, since two consecutive iterations will not result in close 'x' values that can meet the convergence limits. Also, since J is singular, hence the condition numner given by $||J||||J^{-1}||$ will be infinitely large indicating an ill-conditioned system, which will not potentially converge.

3: Chapter-9: Question-13

To minimize the function:

$$\phi(x) = c^T + \frac{1}{2}x^T Hx$$

where, c, H and initial x_0 is provided.

Both Newton and BFGS are implemented in the file, Prob3.m. The convergence criteria considered is $||x_{k+1} - x_k|| < 1e^{-10}$. The Newton's method converges in 2 iterations to the solution $x = [1.000, 0.000, -1.000, 2.000]$. BFGS converges in 20 iterations to the solution $x = [1.000, 0.000, -1.000, 2.000]$. For choosing α in the BFGS method we have implemented the algorithm combining backtracking and weak-line search.

The general converegnce expression for any method is given as:

$$\lim_{n->\infty} \frac{|x_{n+1} - x^*|}{|x_n - x^*|^\alpha} = \lambda$$

where, λ is the asymptotic error constant and α denotes the rate of convergence. For Newtons's method, if the initial guess is relatively close enough to the solution, then the convergence is quadratic, that is, $\alpha = 2$. For BFGS, the convergence is superlinear, that is, $1 < \alpha < 2$. Thus, BFGS is slower to converge than the quadratic convergence of Newton's method, which explains why BFGS takes larger number of iterations .

4: Chapter-9: Question-22

Given problem:

$$\min_y ||y||_p; s.t. Jy = b$$

where J is $m \times n, m \leq n$ and p=2, indicating L2-norm.

(a) Here, we have squared the l2-normed expression, which allows us to get rid of the square-root sign. The multiplication by $\frac{1}{2}$ is just a constant which we can get rid of when taking the first derivative. This does not changes the minimizer because, the solution coefficients obtained will still remain the same. Basically, the point where $||y||_2^2$ is minimized will be the same where $||y||_2$ is minimized.

(b) So, now we have a slightly modified form of the the problem:

$$\min_y \frac{1}{2}||y||_2^2; s.t. Jy = b$$

Now we can break the objective function further as:

$$\frac{1}{2}||y||_2^2 = \frac{1}{2}y^T I y - 0^T y$$

This now becomes a quadratic programming problem with equality constraints. Thus, we have the Lagrangian as:

$$\mathcal{L}(y, \lambda) = \frac{1}{2}y^T I y - \lambda^T (b - Jy)$$

Then the KKT conditions will give:

$$\begin{aligned} \Delta_y \mathcal{L}(y^*, \lambda^*) &= 0 \\ Iy + J^T \lambda &= 0 \end{aligned} \tag{5}$$

and

$$\begin{aligned} c_i(y^*) &= 0 \\ b - Jy &= 0 \end{aligned} \tag{6}$$

Combining the above, we get the following system of linear equations:
$\begin{bmatrix} I & J^T \\ J & 0 \end{bmatrix} \begin{bmatrix} y \\ \lambda \end{bmatrix} = \begin{bmatrix} 0 \\ b \end{bmatrix}$

5: Chapter-10: Question-6

Given 4 data points as below:

$$\begin{aligned} (x_0, y_0) &= (-1, 1) \\ (x_1, y_1) &= (0, 1) \\ (x_2, y_2) &= (1, 2) \\ (x_3, y_3) &= (2, 0) \end{aligned}$$

(Monomial Basis)

$$p(x) = \sum_{j=0}^{3} c_j x^j$$

where c_j are the coefficients of the j'th basis.

At (x_0, y_0),

$$p(x_0) = p(-1) = c_0 - c_1 + c_2 - c_3 = y_0 = 1$$

At (x_1, y_1),

$$p(x_1) = p(0) = c_0 = y_1 = 1$$

At (x_2, y_2),

$$p(x_2) = p(1) = c_0 + c_1 + c_2 + c_3 = 2$$

At (x_3, y_3),

$$p(x_3) = p(2) = c_0 + 2c_1 + 4c_2 + 8c_3 = 0$$

Combining these above 4 equations, we get the following system of equations:

$$\begin{bmatrix} 1 & -1 & 1 & -1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 8 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 2 \\ 0 \end{bmatrix}$$

Solving this system of equation, we get the following coefficients values for monomial basis:

$c_0 = 1$, $c_1 = 1.16667$, $c_2 = 0.5$, $c_3 = -0.6667$

Thus the interpolating cubic polynomial using monomial basis will be:

$$p(x) = 1 + 1.6667x + 0.5x^2 - 0.6667x^3 \tag{7}$$

(Lagrange)

In the Lagrange, the j'th basis function is defined as:

$$L_j(x) = \frac{(x - x_0)\dots(x - x_{j-1})(x - x_{j+1})\dots(x - x_n)}{(x_j - x_0)\dots(x_j - x_{j-1})(x_j - x_{j+1})\dots(x_j - x_n)}$$

and the polynomial is defined as:

$p(x) = \sum_{j=0}^{n} y_j L_j(x) = 1\dfrac{(x-0)(x-1)(x-2)}{(-1-0)(-1-1)(-1-2)} + 1\dfrac{(x+1)(x-1)(x-2)}{(0+1)(0-1)(0-2)} + 2\dfrac{(x+1)(x-0)(x-2)}{(1+1)(1-0)(1-2)} + 0.L_3(x) = \dfrac{-4x^3 + 3x^2 + 7x + 6}{6}$

Hence, we get the following coefficient values for the Lagrange's basis:

$c_0 = 1$, $c_1 = 1.16667$, $c_2 = 0.5$, $c_3 = -0.6667$

Hence, the interpolating cubic polynomial with Lagrange basis will be ,

$$1 + 1.16667x + 0.5x^2 - 0.667x^3 \tag{8}$$

(Newton) For a cubic interpolant, the basis functions will be:
$\phi_0(x) = 1$, $\phi_1(x) = x - x_0 = x + 1$, $\phi_2(x) = (x - x_0)(x - x_1) = (x - x_0)x$, and $\phi_3(x) = (x - x_0)(x - x_1)(x - x_2) = (x + 1)x(x - 1) = (x^3 - x)$

At (x_0, y_0)
$p(-1) = c_0 = 1$

At (x_1, y_1)
$p(0) = 1 + c_1(x - x_0) = y_1 => c_1 = 0$

At (x_2, y_2)
$p(1) = 1 + 0 + c_2(x - x_0)(x - x_1) = y_2 => 1 + c_2(1 + 1)(1 - 0) = 2 => c_2 = \dfrac{1}{2}$

At (x_3, y_3)
$P(2) = 1 + 0 + c_2(2 + 1)(2 - 0) + c_3(2 + 1)(2 - 0)(2 - 1) = 0, => c_3 = \dfrac{-2}{3}$

Hence, reconstructing the polynomial:

$$p(x) = 1 + 0 + \frac{1}{2}x(1 + x) - \frac{2}{3}(x + 1)x(x - 1)$$

$$p(x) = 1 + \frac{7}{6}x + \frac{x^2}{2} - \frac{2}{3}x^3$$

$$p(x) = 1 + 1.1667x + 0.5x^2 - 0.667x^3$$

Hence, the interpolating cubic polynomial with Lagrange basis will be ,

$$1 + 1.16667x + 0.5x^2 - 0.667x^3 \tag{9}$$

As (7), (8) and (9) shows that the three representations gives the same polynomial.

7: Chapter-10: Question-25

(a) Let the quadratic polynomial be

$$p(x) = c_0 + c_1x + c_2x^2 \tag{10}$$

Then , at x=0:

$$p(0) = c_0 = f(0) = 1$$

and

$$p'(0) = c_1 = f'(0) = -1$$

At, x=1:

$$c_0 + c_1 + c_2 = f(1) = 2$$

Solving the above 3 equations, gives

$$c_0 = 1$$

$$c_1 = -1$$

$$c_2 = 2$$

Plugging these coefficients back to (10), we get the following quadratic interpolant:

$$p(x) = 1 - x + 2x^2 \tag{11}$$

By taking the first derivative and the second derivative of (11), we get:

$$p'(x) = -1 + 4x$$

$$p''(x) = 4 > 0$$

Equating the first derivative to 0, we get a stationary point at $x = \dfrac{1}{4}$, and since the second derivative, $p''(x) > 0$, hence this point is a minima. Thus we have found $x^* = \dfrac{1}{4}$. (Answer)

(b) We already found the minima $x^* = \dfrac{1}{4}$ which is $0 < \dfrac{1}{4} < 1$. Since, it is quadratic, it has only a single maxima or minima becuase its first derivative is linear and hence cannot have multiple roots. Since, we have found a minima, hence it has to be the unique minima satisfying $0 < x^* < 1$. For the original function, given it has continuous second order derivative, however nothing is mentioned about its higher derivatives. Now, since $f(1) > f(0)$, there is a point in $[0, 1)$, from where the function f starts to increase. Since, $f'(0) < 0$, hence $x = 0$ is not the point where f achieves a minimum. So, f achieves a minimum at $x^* = 0 + \delta$, where δ is a positive quantity between 0 and 1. Hence, f has a unique minimum at a point $0 < x < 1$ .Since , f has continuous second order derivative in this region, we do not know about its higher order derivatives. If it is quadratic, then this minima is unique for same reasons as the quadratic polynomial. But , if the underlying function is not quadratic, then the uniqueness cannot be asserted since there can be many more minimas in this region.

8: Chapter-11: Question-3

Let $f \in C^3[a, b]$ be given at equidistant points, $x_i = a + ih$, where $i = 0, 1, 2, \ldots, n$ and $nh = b - a$. $f'(a)$ is given as well.

(a) Construct an algorithm for C^1 continuous quadratic interpolation:

$$v(x) = S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2; x_i \leq x \leq x_{i+1}$$

That is an algorithm to determine the 3n coefficients.

Algorithm: The polynomial $S_i(x)$ in the interval $x_i \leq x \leq x_{i+1}$ will match the function value exactly. So, we have our first condition as:

$$S_i(x_i) = a_i = f(x_i)$$

Thus, we have n equations in n variables due to this. This gives us all the n a_i values.

Also, since $S_i(x)$ is valid in the interval $x_i \leq x \leq x_{i+1}$, hence it should satisfy the function value at x_{i+1} as well. Thus we have our next condition as:

$$S_i(x_{i+1}) = f(x_{i+1})$$

$$a_i + b_i(x_{i+1} - x_i) + c_i(x_{i+1} - x_i)^2 = f(x_{i+1})$$

since, $x_{i+1} - x_i = h$, we get:

$$b_i h + c_i h^2 = a_{i+1} - a_i$$

This gives us n equations in 2n variables. In combination with the above we get 2n equations in 3n variables.

Next, since we want the interpolant to be C^1 continuous and the given fucntion is C^3, hence, the first derivative of $S_i(x_i)$ should match the first derivative of the function at x_i. Thus, we get:

$$S_i'(x_i) = f'(x_i)$$

$$b_i = f'(x_i)$$

Also, the derivatives should match at x_{i+1}. Hence,

$$S_i'(x_{i+1}) = f'(x_{i+1})$$

$$b_i + 2c_i(x_{i+1} - x_i) = f'(x_{i+1})$$

$$b_i + 2c_i h = f'(x_{i+1})$$

There is a certain pattern that emerges. If we know the derivative value at x_i, we can use it further to compute the derivative value at x_{i+1}. Since, we need to evaluate all the derivative values exceopt the first one, let us denote the derivative with the variable d_i.
Thus, from the above relations we have the following set of equations:

n equations for

$$a_i = f(x_i)$$

n equations for

$$b_i h + cih^2 - (a_{i+1} - a_i) = 0$$

n equations for

$$b_i - d_i = 0$$

(n-1) equations for

$$b_i + 2c_i h - d_{i+1} = 0$$

1 equations for the derivative given at x=a, that is d_0

$$d_0 = f(a)$$

Thus, in total we have 4n variables with 4n equations which can be solved to get the 3n coefficients $\{a_i, b_i, c_i\}$ and the derivatives d_i.

One more method: Since, we want the piecewise interpolant to be C^1 continuous, hence at the interior point x_i, the first derivative of $S_{i-1}(x)$ should equal the first derivative of $S_i(x)$. Hence, we have the following relation:

$$S_{i-1}(x_i) = S_i(x_i)$$

$$b_{i-1} + 2c_{i-1}(x_i - x_{i-1}) = b_i$$

$$b_{i-1} - b_i + 2c_{i-1}h = 0$$

This will have (n-1) equations since we equate between two consecutive interior points.
Thus, now we can consider the following set of equations:
n equations for

$$a_i = f(x_i)$$

n equations for

$$b_i h + cih^2 - (a_{i+1} - a_i) = 0$$

(n-1) equations for

$$b_{i-1} - b_i + 2c_{i-1}h = 0$$

and 1 equation for the condition that the derivative at $x = a$, $f'(a)$, is given.

$$S_0'(a) = b_0 = f'(a)$$

$$b_0 - f'(a) = 0$$

Thus, now we have 3n equations and 3n variables, which can be solved to get the 3n coefficients $\{a_i, b_i, c_i\}$.

(b) The error estimate for an n-degree interpolate, $p_n(x)$, with respect to the original function, $f(x)$, is expressed as:

$$f(x) - p_n(x) = \frac{f^{(n+1)}(\zeta)}{(n+1)!}\prod_{i=0}^{n}(x - x_i)$$

For piecewise quadratic interpolant , $v(x) = S_i(x)$, we have n=2. Hence, we get,

$$|f(x) - v(x)| \leq (x - x_{i-1})(x - x_i)\max_{a\leq\zeta\leq b}\frac{f^3(\zeta)}{3!} \tag{12}$$

That is the 3rd derivative is bounded by its max value at a point over the entire interval [a,b]. For the term, $(x-x_{i-1})(x-x_i)$, the maximum occurs at the midpoint of the interval , that is, at $\dfrac{x_{i-1}+x_i}{2}$. Thus, we get:

$$|(x - x_{i-1})(x - x_i)| \leq \left(\frac{x_i - x_{i-1})}{2}\right)^2$$

and since, $x_i - x_{i-1} = h$, we get

$$|(x - x_{i-1})(x - x_i)| \leq \left(\frac{h}{2}\right)^2$$

Plugging this back to (12), we get the following error bound:

$$ErrorBound = |f(x) - v(x)| \leq \frac{h^2}{24}\max_{a\leq\zeta\leq b}|f^3(\zeta)|$$

9: Chapter-11: Question-12

Derive a B-spline basis representation for piecewise linear interpolation and for piecewise cubic interpolation:

For (n+1) interpolating points and degree (k-1), the bspline representation is written as:

$$P(u) = \sum_{i=0}^{n} piN_{i,k}(u) \tag{13}$$

where $N_{i,k}$ are the bspline basis functions and (k-1) is the degree of the curve. Hence, for a piece-wise linear interpolant, the degree of the curve is 1, hence $k = 2$. Thus, we have (n+1) interpolating points and k=2.
The idea of bspline formulation us to generate curve that are affected locally by the control points. So, if a point is changed, the curve formulation will change locally, and not cause changes in the entire curve unlike bezier curves. The entire interval of points $\{p_0, p_1, \ldots, p_n\}$ is divided into a sequence of knots $(t_0, t_1, \ldots, t_r)$, such that the effect of a point will have its control in a knot region $[t_{i-1}, t_i]$ and not anywhere else:
The basis function, $N_{i,k}$, is defined as :

$$N_{i,k}(u) = \frac{(u - t_i)N_{i,k-1}(u)}{t_{i+k} - t_i} + \frac{(t_{i+k} - u)N_{i+1,k-1}(u)}{t_{i+k} - t_{i+1}} \tag{14}$$

The base case of the basis functions is given as:

$$N_{i,1}(u) = 1; t_i \leq u < t_{i+1}$$

$$= 0; otherwise$$

This allows for the termination condition of the recursion defined in (14):

In general, the knot values, t_i are defined as:

$$t_i = 0; i < k$$
$$t_i = i - k + 1; k \leq i \leq n$$
$$t_i = n - k + 2; i > n$$

For the piecewise linear case, the value of $k = 2$, hence the above relations translate to:

$$t_i = 0; i < 2$$
$$t_i = i - 1; 2 \leq i \leq n$$
$$t_i = n; i > n$$

Thus , the set of $(n + k + 1) = (n + 3)$ knot values will be:

$$t = (0, 0, 1, 2, \dots, n - 1, n, n)$$

(13) in this case translates to :

$$P(u) = \sum_{i=0}^{n} p_i N_{i,2}(u) = p_0 N_{0,2}(u) + p_1 N_{1,2}(u) + \cdots + p_i N_{i,2}(u) + \cdots + p_n N_{n,2}(u)$$

where, the basis function $N_{i,2}$ in this case will be :

$$N_{i,2}(u) = \frac{(u - t_i) N_{i,1}(u)}{t_{i+2} - t_i} + \frac{(t_{i+2} - u) N_{i+1,1}(u)}{t_{i+2} - t_{i+1}} \tag{15}$$

Here, the base cases of the recursion will become:
$N_{0,1} = 1$ if $t_0 \leq u < t_1$ or $0 \leq u < 0$
$N_{0,1} = 0, otherwise,$
$N_{0,1} = 0$

$N_{1,1} = 1$ if $t_1 \leq u < t2$ or $0 \leq u < 1$
$N_{1,1} = 0$, otherwise

$N_{2,1} = 1$ if $t_2 \leq u < t_3$ or $1 \leq u < 2$
$N_{2,1} = 0$, otherwise

...
$N_{n,1} = 1$ if $t_n \leq u < t_{n+1}$ or $n - 1 \leq u < n$
$N_{n,1} = 0$, otherwise

The above base cases plugged into the recursion of the $eqrefeq : linbasis$ gives the representation for the bspline basis function for piecewise linear. (Answer).

For the case of the piecewise cubic hermiote interpolant, we have a degree 3 interpolant. Hence, (k-1) is 3, thus k=4. Hence, the relation for deriving the knot values will become:

$$t_i = 0; i < 4$$
$$t_i = i - 3; 4 \leq i \leq n$$
$$t_i = n - 2; i > n$$

Thus, the set of $(n+k+1)=(n+5)$ knot values will be:

$$t = (0,0,0,0,1,2,\ldots,n-3,n-2,n-2,n-2,n-2)$$

(13) in this case translates to :

$$P(u) = \sum_{i=0}^{n} p_0 N_{0,4}(u) + p_1 N_{1,4}(u) + \cdots + p_i N_{i,4}(u) + \cdots + p_n N_{n,4}(u) \tag{16}$$

Where the recurssions over the basis functions can be further expanded following the relation(1), to reach the terminal base cases which in these case will be defined as follows:
$N_{0,1} = 1$ if $t_0 \leq u < t_1$ or $0 \leq u < 0$
$N_{0,1} = 0$, otherwise,
So, $N_{0,1} = 0$

$N_{1,1} = 1$ if $t_1 \leq u < t_2$ or $0 \leq u < 0$
$N_{1,1} = 0$, otherwise,
So, $N_{1,1} = 0$

$N_{2,1} = 1$ if $t_2 \leq u < t_3$ or $0 \leq u < 0$
$N_{2,1} = 0$, otherwise,
So, $N_{2,1} = 0$

$N_{3,1} = 1$ if $t_3 \leq u < t_4$ or $0 \leq u < 1$
$N_{3,1} = 0$, otherwise,

$N_{4,1} = 1$ if $t_4 \leq u < t_5$ or $1 \leq u < 2$
$N_{4,1} = 0$, otherwise,

$N_{5,1} = 1$ if $t_5 \leq u < t_6$ or $2 \leq u < 3$
$N_{5,1} = 0$, otherwise,

...

$N_{n,1} = 1$ if $t_n \leq u < t_{n+1}$ or $(n-3) \leq u < (n-2)$
$N_{n,1} = 0$, otherwise,

Now consider (16), the basis functions recursively depends exactly on 4 base cases if we break them down as follows:

$$N_{0,4} : (N_{0,1}, N_{1,1}, N_{2,1}, N_{3,1})$$
$$N_{1,4} : (N_{1,1}, N_{2,1}, N_{3,1}, N_{4,1})$$
$$N_{2,4} : (N_{2,1}, N_{3,1}, N_{4,1}, N_{5,1})$$

$$N_{3,4} : (N_{3,1}, N_{4,1}, N_{5,1}, N_{6,1})$$

and so on,

In the region , $0 \leq u < 1$, only $N_{3,1} = 1$, other $N'_{i,1}s$ are zero. Now , $N_{3,1}$ is present only in the recursive list of the basis functions associated with (p_0, p_1, p_2, P_3). Hence, in the region $0 \leq u < 1$, only these 4 points serve as the control points with local control over the curve. Other points do not have any effect here. Similarly, in the region $1 \leq u < 2$, only $N_{4,1}$ is 1, which is present in the recursive list of the basis functions associated with (p_1, p_2, p_3, p_4). Hence, in the region $1 \leq u < 2$, only these 4 points serve as the local control points.

Thus plugging in the base cases , $N'_{i,1}s$, present in the recursive list of respective basis functions, we obtain the representation of the bspline basis functions. (Answer).

10: Chapter-11: question-15

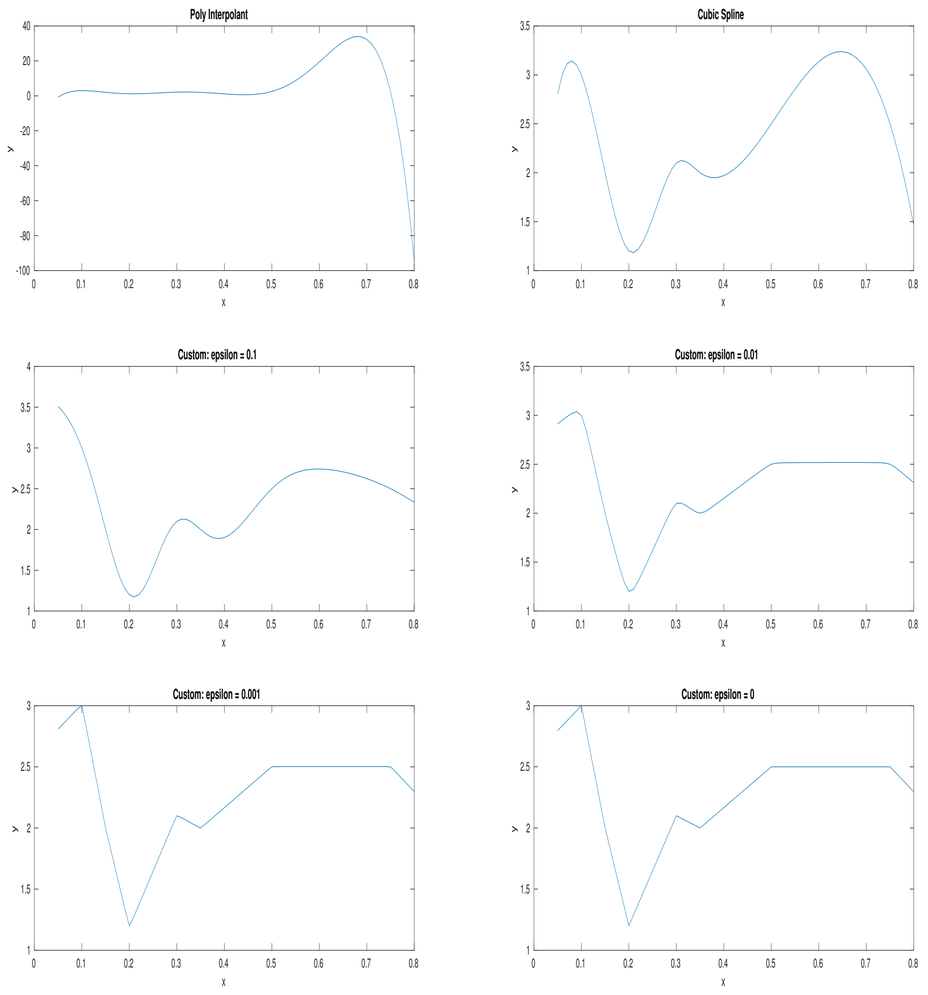

Figure 1: Matlab plots for different interpolant types

The figure shows the plots for the polynomial interpolant, cubic spline and the custom-interpolant method with $\epsilon = 0.1, 0.01, 0.001 and 0$. The matlab code is available as $Prob10.m$. The polynomial interpolant is of degree 6. Although it fits the data correctly, one of its bad property is due to its high degree and global control over all points, around the absiccae points where the data is not available for fitting, like from 0.5 to 0.7, the polynomial has the tendency to oscillate and explode to high values. In comparison, the cubic spline is significantly smooth and the curve is modeled closely to the given interpolating points.Since, the control here is local to set of 4 points, the degree is cubic. For the custom interpolants with the provided basis functions and tunable ϵ, we see that as $\epsilon reduces$, the smoothness of the curve also reduces and around $\epsilon = 0$, it becomes piecewise linear.