AE 370: HW1

Linyi Hou

February 6, 2020

Problem 1

The following are examples of invalid inner products:

$$(u, v) = u(x) + v(x), \forall u, v \in C[1, 2]$$

Not an inner product because (u, v) returns a function and not a scalar.

$$(u, v) = uv - 1, \forall u, v \in \mathbb{R}$$

Not an inner product because $(0, 0) = -1 < 0$

$$(u, v) = u^v, \forall u, v \in \mathbb{Z}$$

Not an inner product because $(\alpha u+\beta v, w) = (\alpha u+\beta v)^w \neq \alpha u^w+\beta v^w = \alpha(u, w)+\beta(v, w)$

Problem 2

2.a

Let $\mathbb{B}$ be a basis for subspace W of the vector space $S = \{f(x) \mid x \in [a, b]\}$. Then,

$$f_a(x) = \sum_{i=1}^{n+1} c_i b_i(x)$$

Consider $n + 1$ coefficients $\{c_1, c_2, ..., c_{n+1}\}$, which require $n + 1$ equations to solve. If we take $n + 1$ points on $[a, b]$: $\{x_1, x_2, ..., x_{n+1}\}$, then

$$f(x_j) = \sum_{i=1}^{n+1} c_i b_i(x_j)$$

Expanding into a linear system $Ac = f$:

$$\begin{bmatrix} b_1(x_1) & b_2(x_1) & \cdots \\ b_2(x_1) & \ddots & \vdots \\ b_{n+1}(x_{n+1}) & \cdots & b_{n+1}(x_{n+1}) \end{bmatrix} \begin{bmatrix} 1 \\ c_2 \\ \vdots \\ c_{n+1} \end{bmatrix} = \begin{bmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_{n+1}) \end{bmatrix}$$

2.b, 2.c

*See MATLAB figures in Appendix A and code in Appendix B.

The condition number is a metric of sensitivity to error for matrix systems. The condition number remains at 1 for the Lagrange basis approximation for all values of n while the condition number for the monomial basis grows exponentially. Hence Lagrange is a better approximation method due to its insensitivity to error at large sample sizes.

2.d

Chebyshev points rely on being able to solve the linear system $Ac = f$ accurately to produce good results. Using the Lagrange basis, the c matrix is easy to compute since it simply contains the values of the sample points $f(x_i)$. Hence the A matrix is simply the identity matrix, which is invertible in the operation $c = A^{-1} * f$.

However, in the monomial basis, computing the inverse of the A matrix becomes difficult since sample points are close for large sample sizes, making certain rows of the A matix almost linearly dependent. This causes the A matrix to be close to singular/non-invertible. A computer program with finite accuracy will thus be prone to error for large samples.

Problem 3

*See MATLAB figures in Appendix A and code in Appendix B.

Appendix A: Figures

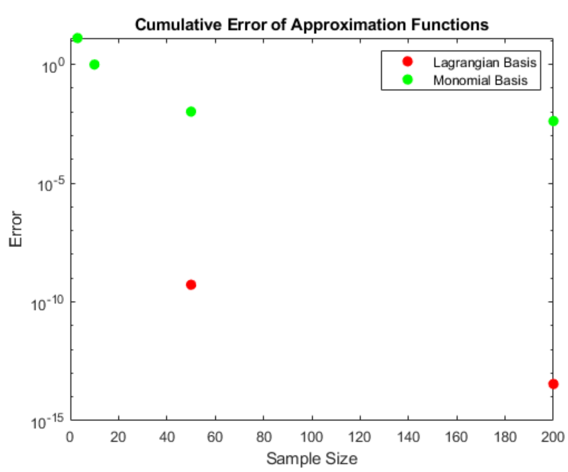

Figure 1: Problem 2.b Error Plots

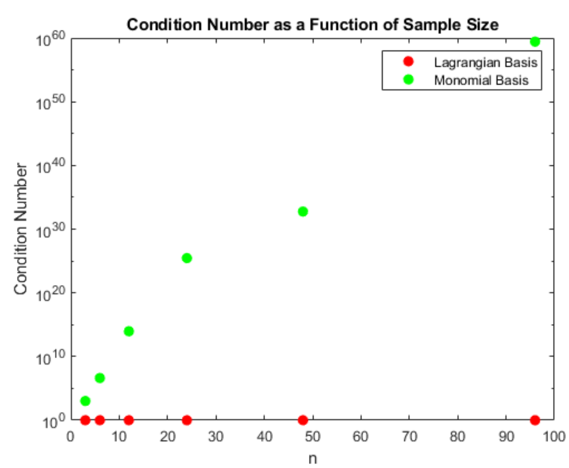

Figure 2: Problem 2.c Condition Numbers

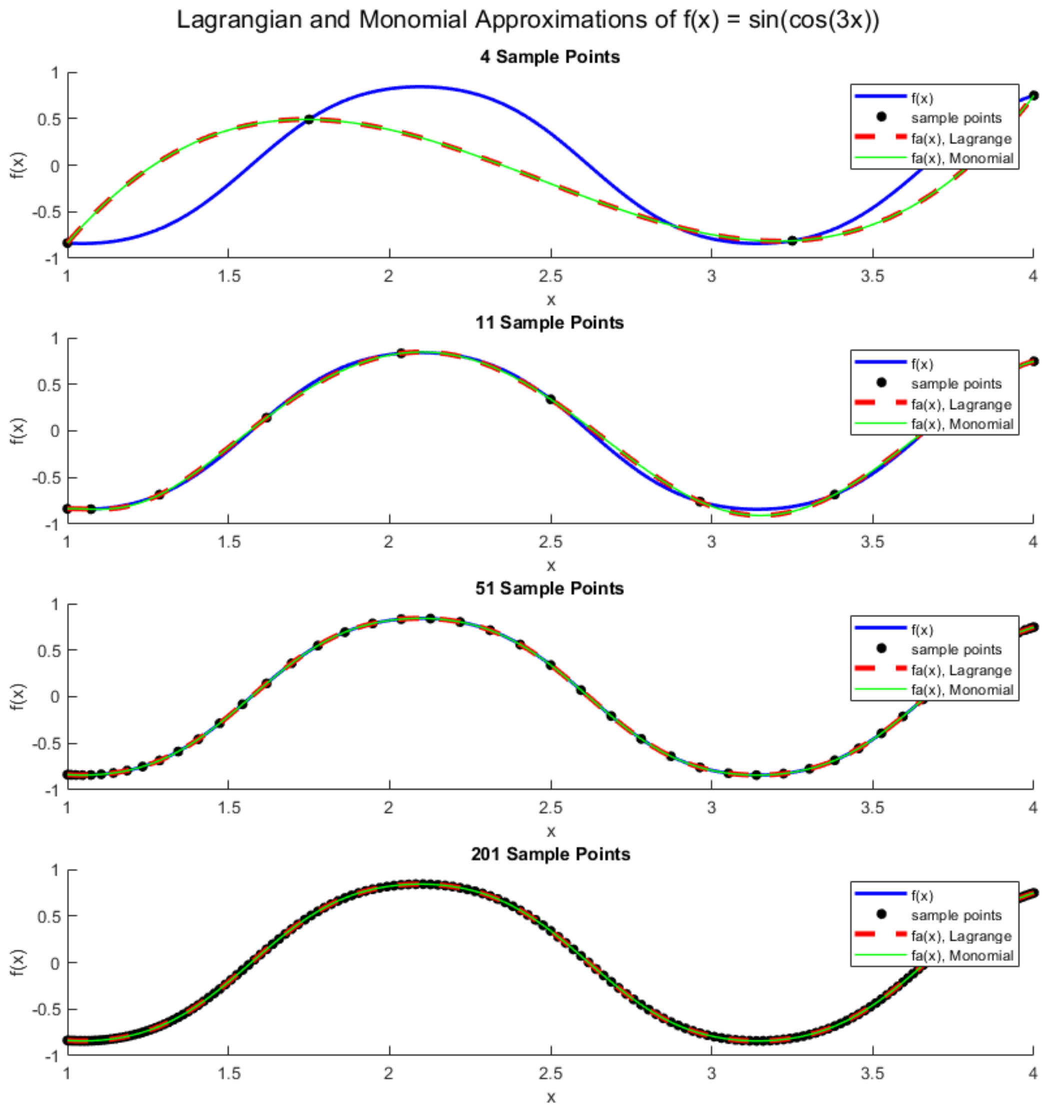

Figure 3: Problem 2.b Function Plots

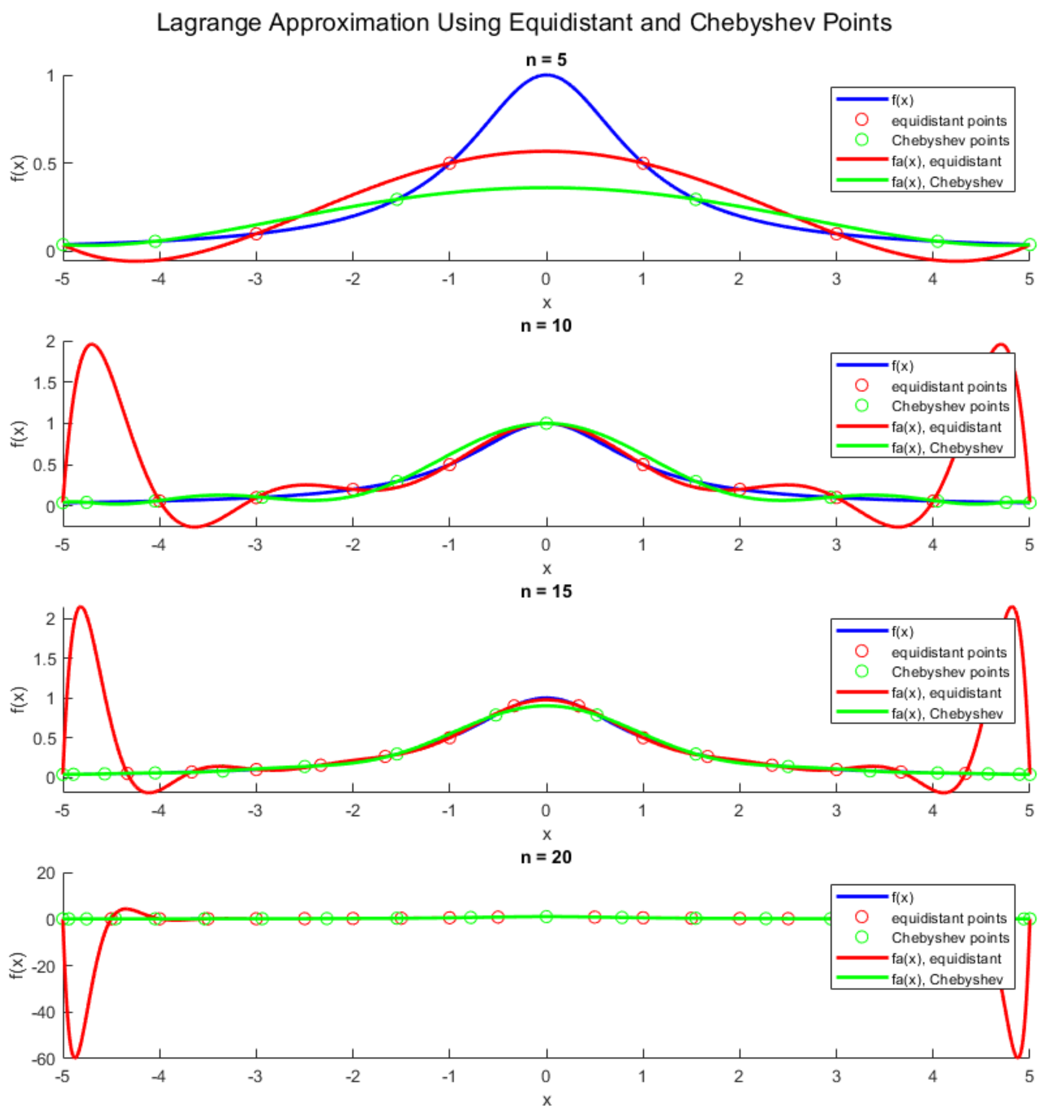

Figure 4: Problem 3.a, 3.b Function Plots

Appendix B: Code

MATLAB Code for Problem 2

```
% --------------------------
% Lagrange basis

n_vect  = [3,10,50,200];
f       = @(x) sin(cos(3*x));
intv    = [1,4];
err_L   = zeros(size(n_vect));

for i = 1:length(n_vect)
    n = n_vect(i);

    xp = (intv(1)+intv(2))/2 + (intv(1)-intv(2))/2*cos((0:n)*pi/n);
    fp = f(xp);
    dp = fp;
    A = eye(n+1);

    xx = linspace(intv(1),intv(2),1000);
    pn_L = 0;
%     L = @(x) 0;
    for j = 1:n+1
        Li = @(x) 1;

        for k = 1:n+1
            if k ~= j
                Li = @(x) (x-xp(k)) ./ (xp(j)-xp(k)) .* Li(x);
            end
        end
%         L = @(x) L(x) + dp(j) .* Li(x);
        pn_L = pn_L + dp(j) * Li(xx);
    end

    figure(1)
    subplot(4,1,i)
    hold on
    plot(xx,f(xx),'b-','lineWidth',2)
    plot(xp,f(xp),'k.','markersize',20)
```

```
    plot(xx,pn_L,'r--','linewidth',3)
    title([num2str(n+1) ' Sample Points'])
    xlabel('x')
    ylabel('f(x)')
    err_L(i) = norm(f(xx)-pn_L);
end

hold off

% ----------------------------------------------------
% Monomial basis

err_M   = zeros(size(n_vect));

for i = 1:length(n_vect)
    n = n_vect(i);

    xp = (intv(1)+intv(2))/2 + (intv(1)-intv(2))/2*cos((0:n)*pi/n);
    fp = f(xp);

    A = zeros(n+1,n+1);
    for j = 1:n+1
        A(:,j) = xp' .^ (j-1);
    end
    dp = A \ (fp');

    xx = linspace(intv(1),intv(2),1000);
    M = @(x) 0;
    for j = 1:n+1
        M = @(x) dp(j) .* x.^(j-1) + M(x);
    end
    pn_M = M(xx);

    figure(1)
    hold on
    subplot(4,1,i)
    plot(xx,pn_M,'g-','lineWidth',1)
    legend('f(x)','sample points','fa(x), Lagrange','fa(x), Monomial')
    title([num2str(n+1) ' Sample Points'])
    xlabel('x')
```

```
    ylabel('f(x)')
    err_M(i) = norm(f(xx)-pn_M);
end

sgtitle('Lagrangian and Monomial Approximations of f(x) = sin(cos(3x))')

hold off

figure(100)
semilogy(n_vect,err_L,'r.','markersize',24,'linewidth',2), hold on
semilogy(n_vect,err_M,'g.','markersize',24,'linewidth',2), hold off
legend('Lagrangian Basis','Monomial Basis')
title('Cumulative Error of Approximation Functions')
xlabel('Sample Size')
ylabel('Error')


% ------------------------------------------
% Part c

n_vect  = [3,6,12,24,48,96];
intv    = [1,4];
condn_L = zeros(length(n_vect),1);
condn_M = condn_L;

% Lagrangian Basis
for i = 1:length(n_vect)
    n = n_vect(i);
    A = eye(n+1);
    condn_L(i) = cond(A);
end

% Monomial Basis
for i = 1:length(n_vect)
    n = n_vect(i);
    xp = (intv(1)+intv(2))/2 + (intv(1)-intv(2))/2*cos((0:n)*pi/n);
    A = zeros(n+1,n+1);
    for j = 1:n+1
        A(:,j) = xp' .^ (j-1);
    end
```

```
    condn_M(i) = cond(A);
end

figure(200)
semilogy(n_vect,condn_L,'r.','markersize',24,'linewidth',2), hold on
semilogy(n_vect,condn_M,'g.','markersize',24,'linewidth',2), hold off
legend('Lagrangian Basis','Monomial Basis')
title('Condition Number as a Function of Sample Size')
xlabel('n')
ylabel('Condition Number')
```

MATLAB Code for Problem 3

```
% -------------------------------
% Lagrange basis

n_vect  = [5,10,15,20];
f       = @(x) 1 ./ (1+x.^2);
intv    = [-5,5];

for i = 1:length(n_vect)
    n = n_vect(i);

    xp_E = intv(1):(intv(2)-intv(1))/n:intv(2);
    xp_C = (intv(1)+intv(2))/2 + (intv(1)-intv(2))/2*cos((0:n)*pi/n);
    fp_E = f(xp_E);
    fp_C = f(xp_C);
    dp_E = fp_E;
    dp_C = fp_C;

    xx = linspace(intv(1),intv(2),1000);
    pn_E = 0;
    for j = 1:n+1
        Li = @(x) 1;

        for k = 1:n+1
            if k ~= j
                Li = @(x) (x-xp_E(k)) ./ (xp_E(j)-xp_E(k)) .* Li(x);
            end
```

```
        end
        pn_E = pn_E + dp_E(j) * Li(xx);
    end

    pn_C = 0;
    for j = 1:n+1
        Li = @(x) 1;

        for k = 1:n+1
            if k ~= j
                Li = @(x) (x-xp_C(k)) ./ (xp_C(j)-xp_C(k)) .* Li(x);
            end
        end
        pn_C = pn_C + dp_C(j) * Li(xx);
    end

    figure(3)
    subplot(4,1,i)
    hold on
    plot(xx,f(xx),'b-','lineWidth',2)
    plot(xp_E,f(xp_E),'ro','markersize',7)
    plot(xp_C,f(xp_C),'go','markersize',7)
    plot(xx,pn_E,'r-','linewidth',2)
    plot(xx,pn_C,'g-','linewidth',2)
    legend( 'f(x)',...
            'equidistant points',...
            'Chebyshev points',...
            'fa(x), equidistant',...
            'fa(x), Chebyshev')
    title(['n = ' num2str(n)])
    xlabel('x')
    ylabel('f(x)')

end

sgtitle('Lagrange Approximation Using Equidistant and Chebyshev Points')

hold off
```