This Lecture

- HW3 answers
- Coding time.
HW3 Q1

Q1) Code up a function (LEGENDRE) which takes a vector $x$ and $m$ as arguments and computes the $m$'th order Legendre polynomial at $x$ using the stable recurrence relation.

A1) We use the following recurrence relationship:

\[
L_0(x) = 1 \\
L_1(x) = x \\
L_{m+1}(x) = \frac{2m+1}{m+1} xL_m(x) - \frac{m}{m+1} L_{m-1}(x), \quad m = 0, 1, \ldots
\]
function leg = LEGENDRE(x, p)

  if(p==0)
    leg = ones(size(x));
  end

  if(p==1)
    leg = x;
  end

  if(p>=2)
    legm1 = ones(size(x));
    leg = x;

    for m=1:p-1
      legp1 = ((2*m+1)*x.*leg - m*legm1)/(m+1);
      legm1 = leg;
      leg = legp1;
    end
  end

end
• Let's plot those Legendre polynomials and see how they behave over the interval [-1,1]

cols = 'bgorxc*msydkv';
N = 200;
x = linspace(-1,1,N);
clf
for p=0:6
    legp = LEGENDRE(x,p);
    hold on;
    plot(x, legp, cols( (p+1)*2:(p+1)*2+1));
    hold off;
end
HW3 Q2

Q2) Create a function (LEGdiff) which takes \( p \) as argument and returns the coefficient differentiation matrix \( D \)

A2) The differentiation matrix which maps the Legendre coefficients to the Legendre coefficients of the derivative of the function:

\[
\hat{D}_{mj} = \begin{cases} 
(2m + 1) & \text{if } j > m \text{ and } j + m \text{ odd} \\
0 & \text{otherwise}
\end{cases}
\]
HW3 Q2 cont

A2cont) The Matlab code is:

```matlab
function Dhat = LEGdiff(p)

    Dhat = zeros(p+1);

    for n=0:p
        for m=n+1:2:p
            Dhat(n+1,m+1) = (2*n+1);
        end
    end
```

HW3 Q2 cont

• Let’s test the output for \( p=5 \):

\[
\hat{D}_{mj} = \begin{cases} 
(2m+1) & \text{if } j>m \text{ and } j+m \text{ odd} \\
0 & \text{otherwise}
\end{cases}
\]

Spot on!.

Note that there are only \( p + (p-2) + (p-4) \).. Non-zero entries
HW3 Q2 cont

- The form of Dhat

```matlab
>> Dhat = LEGdiff(25);
>> spy(Dhat)
>>
```
HW 3 Q3

Q3) Create a function (LEGmass) which takes \( p \) as argument and returns the mass matrix \( M \)

A3) Definition of the mass matrix:

\[
M_{nm} = \begin{cases} 
  \frac{x_{i+1} - x_i}{2} & (n=m) \\
  \frac{2}{2n+1} & (n \neq m)
\end{cases}
\]

```
function mass = LEGmass(p)

mass = diag(2./(2*(0:p)+1));
```

(C:\Documents and Settings\Tim\Desktop\LEGmass.m)
HW3 Q3 cont

\[ \text{ans} = \begin{bmatrix} 2.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.6667 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.4000 & 0.2857 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0.2222 & 0.2222 & 0.2222 & 0.0000 & 0.0000 & 0.0000 \\ 0.1818 & 0.1818 & 0.1818 & 0.1818 & 0.0000 & 0.0000 \\ 0.1538 & 0.1538 & 0.1538 & 0.1538 & 0.1538 & 0.0000 \end{bmatrix} \]

\[ M = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \]
HW3 Q4

Q4) Create a function LEGvdm which returns the following matrix:

A4) \[ V_{ij} = L_j(x_i) \quad 0 \leq i, j \leq p \]
HW3 Q4 cont

\[ V = \begin{bmatrix} 1.0000 & -1.0000 & 1.0000 \\ 1.0000 & 0.0000 & 0.5000 \\ 1.0000 & 1.0000 & 1.0000 \end{bmatrix} \]
HW3 Q5

Q5) Test them with the following matlab routine:

```matlab
p = 7;
x = transpose(sin(0.5*pi*linspace(-1,1,p+1)));
V = LEGvdm(x,p);
Dhat = LEGdiff(p);
M = LEGmass(p);

for m=1:8
F = x.^m;
Fcoeff = V\F;
diffFcoeff = (Dhat*Fcoeff);
diffF = V*diffFcoeff;
error = max(abs((m)*x.^(m-1) - diffF));
end
```
HW3 Q5 cont

• Note that $x^m$ is exactly interpolated for $m \leq p$ when we use $p+1$ points and up to $p$'th order polynomials. $x^{(p+1)}$ is only approximately interpolated – and hence the derivative is only approximate.

```matlab
p = 7;

x = transpose(sin(0.5*pi*linspace(-1,1,p+1)));

V = LEGvdm(x,p);

Dhat = LEGdiff(p);

M = LEGmass(p);

for m=1:8
    F = x.^m;
    Fcoeff = V\F;
    diffFcoeff = (Dhat*Fcoeff);
    diffF = V*diffFcoeff;
    error(m) = max(abs((m)*x.^(m-1) - diffF));
end

plot(error, 'r*'); hold on; plot(error, 'k-'); hold off;
```
Note on Q5

• If we were to repeat this on an interval 
  
  \[ [x_n, x_{n+1}] \]

• We would need to use:

  \[
  \frac{dF}{dx} = \left( \frac{2}{x_{n+1} - x_n} \right) \hat{V}D\left( V^{-1}F \right)
  \]
HW3 Q6

Q6) Compute the condition number of the generalized Vandermonde matrix constructed with p=0,…40 and p+1 Chebychev points.

```matlab
for p=0:140
    x = sin(0.5*pi*linspace(-1,1,p+1))';
    V = LEGvdm(x, p);
    cgraph(p+1) = cond(V);
end
plot(cgraph, 'r**');
hold on; plot(cgraph, 'k-'); hold off;
xlabel('p-order'); ylabel('cond(V)');
```
Asymptotic Behavior $\text{cond}(V)$ as $p \to \infty$

```matlab
>> plot(cgraph./(1:p+1).^2, 'r-')
>>
```

![Plot of \( \text{cond}(V) \) vs. \( p \)](image)
Cliff note version of DG advection equation
Workshop Code

• Using your own routines or those on the web site code up the DG advection equation solver by Wednesday end of class.

• Spare time this class for coding…