Julia Dohner
February 26, 2018
SIOC 221B

HW #2

Problem 1

a)

i. To calculate the coefficient of each function in the fit, we first set up the matrices for the expression Gm=d (one set for u, another for v):

$$For\,u\colon \begin{bmatrix}1 & 10 & 0\\ 1 & 0 & 10\\ 1 & -10 & 0\\ 1 & 0 & -10\\ 1 & 0 & 0\end{bmatrix}\begin{bmatrix}b_{0u}\\ b_{1u}\\ b_{2u}\end{bmatrix} = \begin{bmatrix}35.5\\ 53.5\\ 49.2\\ 33.5\\ 43.5\end{bmatrix} \quad , \quad For\,v\colon \begin{bmatrix}1 & 10 & 0\\ 1 & 0 & 10\\ 1 & -10 & 0\\ 1 & 0 & -10\\ 1 & 0 & 0\end{bmatrix}\begin{bmatrix}b_{0u}\\ b_{1u}\\ b_{2u}\end{bmatrix} = \begin{bmatrix}35.5\\ 53.5\\ 49.2\\ 33.5\\ 43.5\end{bmatrix}$$

We then solve for the coefficients in the m matrices by evaluating the following:

$$m_u = (G^T G)^{-1} G^T du \quad (Equation\ (17) from\ LSF\ notes)$$
$$m_v = (G^T G)^{-1} G^T dv$$

The results are as follows:

$$m_u = \begin{bmatrix}43.0400\\ -0.6850\\ 1\end{bmatrix} \quad m_v = \begin{bmatrix}24.0800\\ -0.1750\\ 0.6000\end{bmatrix}$$

ii. The values of velocity at each instrument are achieved by the following process

$$vel_u = G * m_u = \begin{bmatrix}36.1900\\ 53.0400\\ 49.8900\\ 33.0400\\ 43.0400\end{bmatrix} \quad vel_v = G * m_v = \begin{bmatrix}22.3300\\ 30.0800\\ 25.8300\\ 18.0800\\ 24.0800\end{bmatrix}$$

iii. $Misfit = \mathcal{E} = e^T e = (Gm_u - d_u)^T (Gm_u - d_u)$

$For\,u\colon\ \mathcal{E} = 1.5870 \quad , \quad For\,v\colon\ \mathcal{E} = 4.3430$

iv. $u = b_{0u} + b_{1u}x + b_{2u}y \quad , \quad v = b_{0v} + b_{1v}x + b_{2v}y$

$$\frac{du}{dx} = b_{1u} \quad \frac{du}{dy} = b_{2u} \quad \frac{dv}{dx} = b_{1v} \quad \frac{dv}{dy} = b_{2v}$$

$$vorticity = \frac{dv}{dx} - \frac{du}{dy} \quad divergence = \frac{du}{dx} + \frac{dv}{dy} \quad area\ averaged\ velocity = b_{0u} + b_{0v}$$

$$vorticity = b_{1v} - b_{2u} = -1.1750$$
$$divergence = b_{1u} + b_{2v} = 0.0850$$
$$aav = b_{0u} + b_{0u} = 67.12$$

v. See MATLAB code for calculation of covariance matrix.

$$covariance = \begin{bmatrix} 1.8 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.0450 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.0450 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.8000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0450 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.0450 \end{bmatrix}$$

vi. $variance\ in\ \alpha$: $< \alpha^{12} > = \sigma^2 b^T G^{-g} G^{-gT} b$

See MATLAB code for process and b vectors.

Variance in vorticity = 0.0900
Variance in divergence = 0.0900
Variance in area-averaged velocity = 3.6000

b)

i. We enforce the constraint that divergence = 0 by asserting Fm=h, where F = [0 1 0 0 0 1] and h = 0.

$$m_{constrained} = \begin{bmatrix} 43.0400 \\ -0.6425 \\ 1 \\ 24.0800 \\ -0.1750 \\ 0.6425 \end{bmatrix}$$

ii. $$d_{calc,constrained} = \begin{bmatrix} 36.6150 \\ 53.0400 \\ 49.4650 \\ 33.0400 \\ 43.0400 \\ 22.3300 \\ 30.5050 \\ 25.8300 \\ 17.6550 \\ 24.0800 \end{bmatrix}$$

iii. $\varepsilon = 6.6525$

iv. Vorticity = -1.1750
Divergence = 0
Area-averaged velocity = 67.1200

v. We solve for the constrained covariance matrix using the following expressions:

$$G^{-g} = I - (G^T G)^{-1} F^T (F * (G^T G)^{-1} F^T) * (G^T G)^{-1} G^T$$
$$covariance = G^{-g} \sigma^2 I G^{-g}$$

$$covariance = \begin{bmatrix} 1.8 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.0225 & 0 & 0 & 0 & -0.0225 \\ 0 & 0 & 0.0450 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.8000 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0450 & 0 \\ 0 & -0.0225 & 0 & 0 & 0 & 0.0225 \end{bmatrix}$$

vi. Variance in constrained vorticity = 0.0900
Variance in constrained divergence = 0
Variance in constrained area-averaged velocity = 3.6000

We can see from the results that there are off-diagonal values in the covariance matrix of the constrained version of the calculations, indicating that there is dependence between terms (in this case between du/dx and dv/dy, which comprise the divergence). We can also see that the error in divergence in the constrained version is 0 (in contrast to the unconstrained version, where its variance is 0.0900) because we have forced divergence to equal 0 and not vary.

Problem 2

1. To solve for the transports of each water mass, we set up the expression Gm=d where G is a matrix containing the scalars in front of each transport term in equations 1-4, m is a matrix of the transports, and d is a matrix containing the values in the right-hand sides of each of the equations. From this calculation, we get that the transports are 3.8048e6 for the NADW, 9.2975e6 for the AABW, 2.5365e6 for the PIW and 1.5639e7 for the CIW (all in m^3s^{-1}).

   The proportions of the different water masses making up the Common Water are as follows: 0.5945 of AABW, 0.2433 of NADW, and 0.1622 of PIW. The residence time of the Common Water is 3.8366e10 seconds, or 1216.6 years.

2. Assuming that the radiocarbon data are so unreliable that we must neglect equation 4, the smallest model in the L_2 norm with non-negative transports is:

$$m_{L2} = \begin{bmatrix} 43.01.1040e-5 \\ 2.5333e6 \\ 0 \\ 2.5333e6 \end{bmatrix}$$

   which has been achieved by bumping up the m matrix using some proportion of the null space (full calculations are included in the MATLAB code).

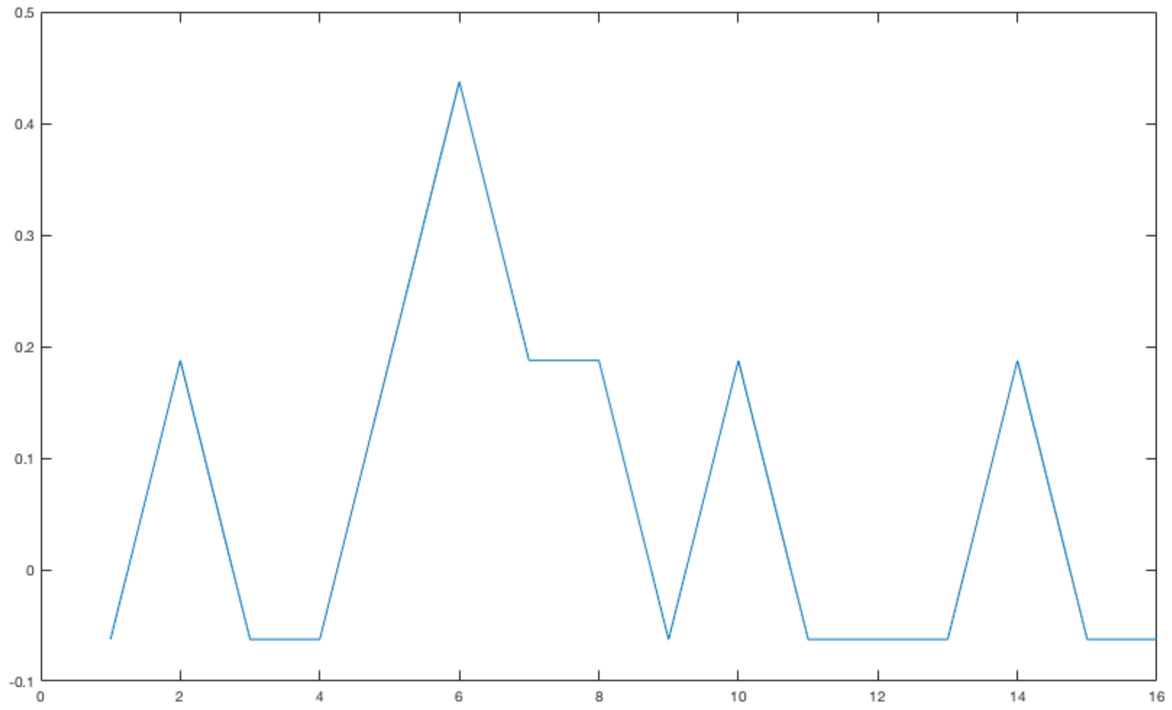
   The new proportions of the water masses comprising the Common Water are simply that the CW consists entirely of AABW. The new residence time of the Common Water is 2.36842e11 seconds, or 7510.21 years.

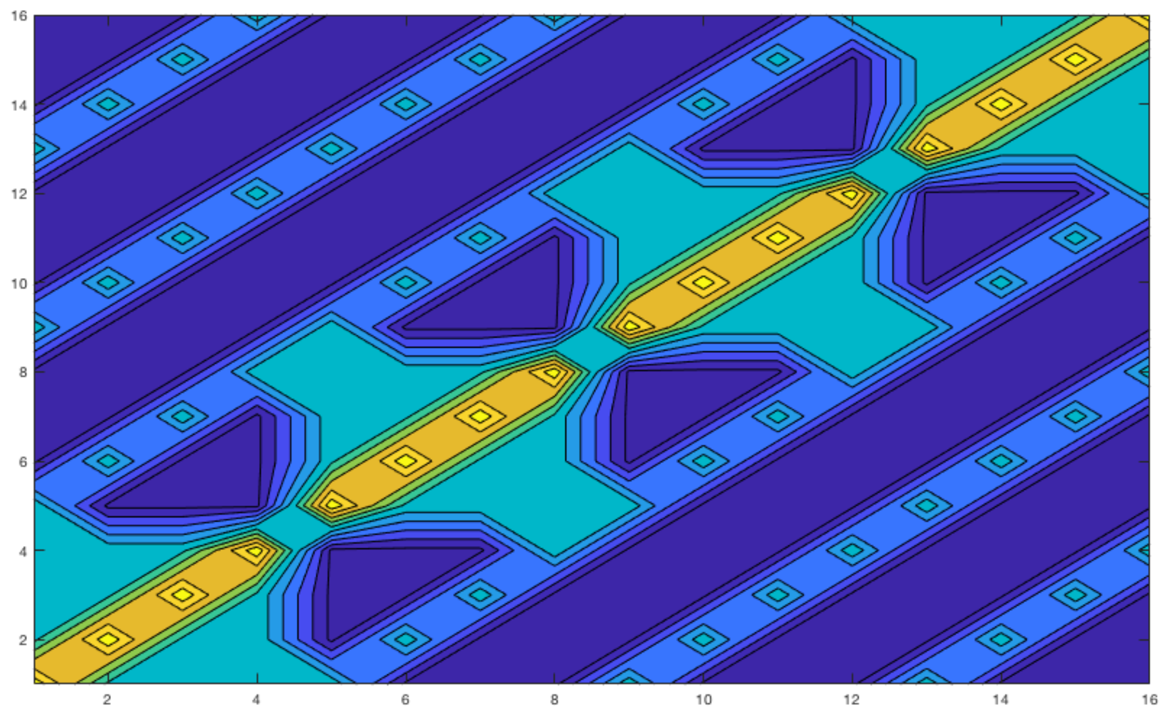
3.
   a. To calculate the solution for the data in the vector da, I first set up a matrix G as follows:

$$G = \begin{bmatrix} 1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&1&1&1&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1 \\ 1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0 \\ 0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0 \\ 0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0 \\ 0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1 \end{bmatrix}$$

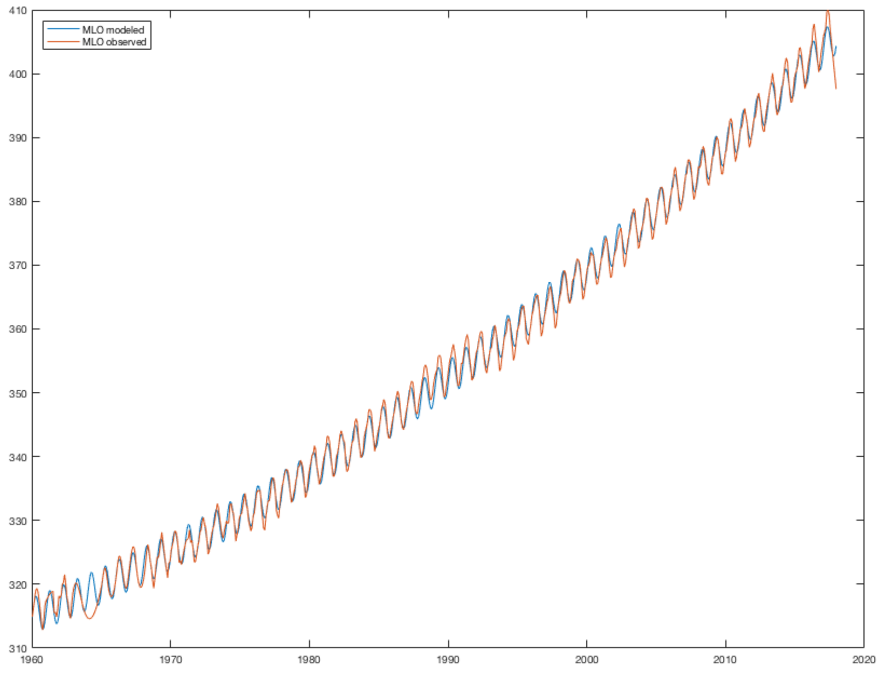
Next I calculated m for a, then b, using the pseudoinverse command in MATLAB (i.e. m_a = pinv(G)*da).

b. I used the approach outlined in part a for my treatment of the data in the vector db.
c. See MATLAB code for approach. R is a 16x16 matrix that yields the following plot (when plotting the row corresponding to the largest slowness in my solution for data db). The second figure is the result of plotting the contour of R.





Individual data calculations, done using the Scripps CO_2 time series of monthly-averaged flask-collected CO_2 samples at the Mauna Loa station. The modeled parameters are a linear trend, an exponential trend, and an oscillatory pattern represented by sines and cosines to emulate the seasonal cycle.



```
% HW 3
% SIOC 221B
% Feb 21, 2018
% Julia Dohner

%% question 1

x = [10 0 -10 0 0];
y = [0 10 0 -10 0];
u = [35.5 53.5 49.2 33.5 43.5]';
v = [21.3 30.4 24.8 18.4 25.5]';
G = [ones(5,1) x' y'];

% i - calculate the coefficient of each function in the fit
m_u = inv(G'*G)*G'*u;
m_v = inv(G'*G)*G'*v;

% ii - values of velocity at each instrument from the plane fit
calc_u = G*m_u;
calc_v = G*m_v;

% iii - the misfit as measured by the L2 norm
% misfit = Gm-d' - Gm-d
misfit_u = (G*m_u-u)'*(G*m_u-u);
misfit_v = (G*m_v-v)'*(G*m_v-v);

%iv - the vorticity, divergence, and area-averaged velocity

vort = m_v(2) - m_u(3);
div = m_u(2) + m_v(3);
aav = m_u(1) + m_v(1); % area averaged velocity "is just b0" - which↙
b0?

%v - covariance matrix

d_co = [u ; v];
m_co = [m_u ; m_v]; % model parameters matrix
G_co = [G zeros(5,3); zeros(5,3) G];
Gco_g = (inv(G_co'*G_co)*G_co'); % G^-g in DR notes
% solution for model parameters in form m_co = Gco_g*d_co + v (eqn↙
50)
```

```
sigma = 3;
cov = Gco_g*(sigma^2*eye(10))*(Gco_g)'; % eqn 52 where sigma^2*I is↙
<d'd'T>

% vi - calcaulte resulting error in vort, div and aav
% equation 56 in LSF notes

% variance in alpha given the data covariance matrix (eqn 57)
% vorticity, divergence and aav are all scalar quantities alpha =↙
beta(m)
% question: what is the b vector in equation 55?
% b = [];
b_vort = [ 0 0 -1 0 1 0]';
b_div = [0 1 0 0 0 1]';
b_aav = [1 0 0 1 0 0]';
var_vort = sigma^2*b_vort'*Gco_g*Gco_g'*b_vort;
var_div = sigma^2*b_div'*Gco_g*Gco_g'*b_div;
var_aav = sigma^2*b_aav'*Gco_g*Gco_g'*b_aav;

% part b - enforce constraint that flow has - divergence

% i constr - calculate coefficient of each function in the fit (find↙
m)
F = [0 1 0 0 0 1];
h = 0;
% solution to constraint given by equation 25:
m_cons = inv(G_co'*G_co)*(G_co'*d_co-F'*inv(F*inv(G_co'*G_co)*F')...
    *(F*inv(G_co'*G_co)*G_co'*d_co-h));

% ii constr - values of velocity at each instrument from the plane↙
fit
calc_cons = G_co*m_cons;

% iii constr - misfit from L2 norm
% misfit = Gm-d' - Gm-d
misfit_cons = (G_co*m_cons-d_co)'*(G_co*m_cons-d_co);

%iv constr - the vorticity, divergence, and area-averaged velocity
vort_cons = m_cons(5) - m_cons(3);
div_cons = m_cons(2) + m_cons(6); % m_cons goes [b0u;b1u;b2u;b0v;b1v;↙
b2v]
aav_cons = m_cons(1) + m_cons(4);
```

```
% v constr - covariance matrix
Gcons_g = (eye(6,6) - inv(G_co'*G_co)*F'*inv(F*inv(G_co'*G_co)*F')*F)↙
*inv(G_co'*G_co)*G_co';
cons_cov = Gcons_g*(sigma^2*eye(10))*(Gcons_g)';


% vi - calcaulte resulting error in vort, div and aav
% equation 56 in LSF notes

% variance in alpha given the data covariance matrix (eqn 57)
var_vort_cons = sigma^2*b_vort'*Gcons_g*Gcons_g'*b_vort;
var_div_cons = sigma^2*b_div'*Gcons_g*Gcons_g'*b_div;
var_aav_cons = sigma^2*b_aav'*Gcons_g*Gcons_g'*b_aav;
```

```
% HW 3 question 2

%% question 2

clear all

% Gm = d
% model params are transports

Ti = [2.5 0.0 4.0 -1.5];
Si = [34.9 34.7 34.4 -34.7];
Ci = [0.90 0.88 0.87 -0.77];

G = [1 1 1 -1; Ti ; Si ; Ci ];
d = [0 -3.8e6 0 2.3e6*0.77]';

% use linsolve to solve AX = B for the vectors of unknowns X
X = linsolve(G,d);

% calculate the coefficient of each function in the fit
% solve for the transports of each water mass
m = inv(G'*G)*G'*d;

% results physically meaningful (all non-negative, where u1 u2 and u3↙
are
% incoming, and u4 is outgoing in equation 1

% proportions of water masses
CW = m(4);
NADW_frac = m(1)/CW;
AABW_frac = m(2)/CW;
PIW_frac = m(3)/CW;

% residence time
CW_volume = 6e17;
tau = CW_volume/CW; % in seconds

% part 2


% 3 data, 4 unknowns- underdetermined proble, did in class, smallest↙
model
% in L2 norm)
%
% will have null space, can add any part of null space and satisfy↙
the
% equation
%
% want all transports to be positive, add some something null space,↙
find
% some of the null space via singular value decomposition- use MATLAB
% command
%
% can add in some of null space to bump up the transport so that↙
they're
% all positive and physically reasonable

G_cut = G(1:3,:);
d_cut = d(1:3);
m_cut = G_cut'*inv(G_cut*G_cut')*d_cut; % smallest model in L2 norm↙
(minimized)
% transports are not physical because some of them are negative

% can the ui be made non-negative while still satisfying the↙
equations 1-3?
% null space of G: subspace consisting of all solutions to Gm = 0
null_G = null(G_cut);
const = 0;
ratio = m_cut./null_G;
min = min(ratio);
%G_bump = G_cut + null_G.*const;
m_bump = m_cut-min.*null_G;

% proportions of different water masses
CW_und = m_bump(4);
NADW_frac_und = m_bump(1)/CW_und;
AABW_frac_und = m_bump(2)/CW_und;
PIW_frac_und = m_bump(3)/CW_und;

% residence time CW
tau_und = CW_volume/CW_und;
```

```
% hw 3
% julia dohner
% question 3

load data1.mat;

% 8 datum (sound speed thru each row and column)
% 16 model parameters
% underdetermined problem
% answer with smallest model size with L2 norm
% 16 parameters, 8 unknowns
% pinv to find

% d = Gm_real answer
% m by m matrix, every row is approximation of delta function


% G is 16x8 matrix
% d is 8x1 (da or db)
% m is 16x1 of slowness parameters
G = [ 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0; % summing first column of↙
slowness vals
    0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 0; % summing second column of↙
slowness vals etc etc
    0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0;
    0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1;
    1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0; % same but thru rows
    0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0;
    0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0;
    0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 ];

% calculate m
%m_a = G'*inv(G*G')*da;
%m_b = G'*inv(G*G')*db;
m_a = pinv(G)*da;
m_b = pinv(G)*db;

% sing prec means here that rows aren't all linearly indepdentn, not↙
all
% giving new info in the end, lin dep because can make one of the↙
rows from the
% others. Have one too many conditions, one is redundant
% stabilization- can add value of lambda, tiny portion
% pseduo-inverse will work

% calculate resolution for matrix
G_g = pinv(G);%m_a; %(inv(G'*G)*G'); %G_g is whatever is multiplying↙
the data
% a lot of different solutions, one is constraint, etc, long and↙
torturous
% for part 1
R = G_g*G;
% each one of rows of R is the approximation to a delta fx I get from↙
the
% solution, can take a row 9 would be approx of delta function to↙
position
% 9 on the grid from the hw problem-- biggest at column 9, then get↙
smaller

% i don't have perfect resolution because i dont have enough data to↙
know
% it perfectly/have perfect resolution

%plot
figure
plot(1:size(R,1),R(6,:));
figure
contourf(R)
```

```
% hw 3
% Julia Dohner
%
% calcs to try with my data
% using keeling co2 data (flask-sampled monthly averages) at mlo
% code taken from A_SIOC_221/HW9/Dohner_SIOC221A_HW9_monthly.m

clear all; close all;

%% load CO2 data

dataMLO = fopen('monthly_data/monthly_flask_co2_mlo_JLD.txt');

valsMLO = textscan(dataMLO, '%f %f', ...
    'delimiter','\t');

fclose(dataMLO);

% format of .txt files is year, co2 value
MLOyear = valsMLO{1};
MLOco2 = valsMLO{2};

%MLOco2 = detrend(MLOco2);


% remove flagged data
for i = 1:length(MLOco2)
    if MLOco2(i) == -99.99
        MLOco2(i) = nan;
    end
end

% remove nan's
addpath↙
('/Users/juliadohner/Documents/MATLAB/A_SIOC_221/HW9/Inpaint_nans/Inp↙
aint_nans');
MLOco2 = inpaint_nans(MLOco2);


% plot timeseries
figure('name','Atmospheric CO2 Timeseries');
plot(MLOyear,MLOco2, '.-');
```

```
xlabel('\fontsize{14}year')
ylabel('\fontsize{14}ppm')
title('\fontsize{16}MLO Atmospheric CO2 Record')
legend('\fontsize{12}Mauna Loa','location','northwest');


% calculate mean and a few moments of variables

% 0th moment - mean xco2
mean = mean(MLOco2);

% 1st moment - variance xco2
variance = var(MLOco2);

% 2nd moment - skewness xco2
skew = skewness(MLOco2);

% calculate the standard error

sigma = variance^0.5;
stderr = sigma/(length(MLOco2));

% spectrum of data
% three overlapping segments:
N = length(MLOco2);
Nseg = 4; % number of segments splitting data into
segment_length = N/Nseg; % length of each chunk of data (aka segment↙
length)
M = segment_length/2;

MLO_use=[reshape(MLOco2,segment_length,Nseg)];
MLO_ft=fft(detrend(MLO_use).*(hann(segment_length)*ones(1,Nseg)));
MLO_spec=sum(abs(MLO_ft(1:M+1,:)).^2,2)/N;
MLO_spec(2:end)=MLO_spec(2:end)*2;
% plot spectra
frequency=(1:M+1)/(segment_length/12);
frequency = frequency';
figure('name','Power Spectra of CO2 Records');
loglog(frequency, MLO_spec, '-b')
xlabel('\fontsize{14}cycles per year')
ylabel('\fontsize{14}ppm^2/cpy')
title('\fontsize{16}Power Spectra of CO2 Record')
```

```
legend('\fontsize{12}Mauna Loa Station', 'Location','northeast');


% fitting trend plus annual cycle
t_MLO = MLOyear-MLOyear(1);
cosMLO = cos(2*pi*MLOyear/(1));
sinMLO = sin(2*pi*MLOyear/(1));
G = [ones(length(MLOyear),1) t_MLO sinMLO cosMLO t_MLO.^2];
m = inv(G'*G)*G'*MLOco2;

d_calc = G*m;
plot(MLOyear,d_calc,MLOyear,MLOco2);
legend('MLO modeled','MLO observed', 'Location','Northwest')
```