```matlab
%   This is Definitions_uncoupled.m
%   This script provides the constants to
%   then solve for the time dependent solution of sulfur and carbon
in the whole ocean
%   based on equations that have:
%   1. CaCO3 output based on a first order rate law with DIC
%   2. The Org C output value based on 20% burial fraction for
modernish
%   values
%   3. The reduced sulfur output is tied to the organic carbon
output via
%   the classic Berner ratio of 1-3 wt %
%   4. The oxidized sulfur output (evaporites) is a constant, fixed
to be steady state at %   the beginning of the run (steady state
version based on a first order
%   rate law with [SO4].)
%   5. using bulk carbonate as the overall ocean d13C, and input
data for the change in
%   d13C.
%   6. The organic carbon burial prior to 52.6 Ma is arbitrarily
chosen to
%   fit the data. it basically tells us that the carbon and sulfur
isotopes
%   aren't coupled, but nothing much more profound than that.

%   Give each variable_box # its own number for clarity in ode
vector
%   calls


S_32 = 1;
S_34 = 2;
C_12 = 3;
C_13 = 4;


VolOcean = 1.38e18;         % Volume of the oceans (m^3)

%%%%%%%%%%%%%%%%   Now declare some key variables for the whole
Sulfur Isotope system

%   Isotope fractionation factors

%del_in_S = 4;                   % delta value of SO4 into ocean in
permil
%eps_S_red = -32.6; % offset between ox and reduced S in permil
stdratio_S = 0.0441626;     % 32/34 ratio in VCDT;
del_in_C = -2.5;                % isotopic value of incoming carbon
in permil
eps_C = -27;                % offset between ox and reduced carbon
(epsilon_) in permil
stdratio_C = 0.0112372;     % 13/12 ratio in VPDB
delS_0 = 20.2;
d13C_offset = 1;  %offset between seawater d13C and foram d13C-
```

assumed constant

```
%   Input/Weathering Fluxes

% Stot_0 =2.8;
% F_32_in = 1.5e12;                         % Input of S-32
SO4 to ocean (mol/yr)
% F_34_in = ((del_in_S./1000)+1)*stdratio_S.*F_32_in;   % Input of
S-34 SO4 to ocean (mol/yr)

tau_dic = 100000;          % residence time of DIC in years
dic = 2000e-6;%1760e-6;              % DIC concentration in moles/
kg

F_Ctot_in = (dic.*1029.*VolOcean)./tau_dic;              % carbon
input flux (inventory/res time) in mol/yr


F_12_in = F_Ctot_in./(1+stdratio_C.*((del_in_C./1000)+1));
F_13_in = ((del_in_C./1000)+1)*stdratio_C.*F_12_in;   % Input of
C-13 to ocean (mol/yr)

%   Burial Rates

%F_C12_red = 3.41e12;    % Flux of reduced carbon output in mol/yr

%CS_red_ratio = 3.*32./12;       % C/S output ratio in mol/mol for
reduced phases (based on Berner ratio of 3 in wt %)

%k_evap = 2.25e-8;              % first order rate constant for
evaporite burial in /yr.



%%%%   Now set up the system to be able to have time dependent
parameters


rampstart = 1e7;    % time of step change in F in
rampend = 3e7;
timestep = 0.1e6;
runtime = 76e6;             % run length in years
timearray = 0:timestep:runtime;     % time arrary to pair with F_red
array


del_C_foram_start = xlsread('d13C input data.xlsx',3,'B17');
del_C_foram_ramp =  xlsread('d13C input data.xlsx',3,'B18:B217');
del_C_foram_end = xlsread('d13C input data.xlsx',23,'B217');%

del_C_for_py_start = xlsread('d13C input data.xlsx',3,'E17');
del_C_for_py_ramp =  xlsread('d13C input data.xlsx',3,'E18:E217');
del_C_for_py_end = xlsread('d13C input data.xlsx',3,'E217');%
```

```matlab
F_C12_red_start = F_12_in.*(((del_C_foram_start + d13C_offset)−
del_in_C)./−eps_C); %this is calculated from assuming initial
isotopic steady state
% Findin=dbcFc + dorgForg and rearranging for a fractional organic
carbon
% burial.  I'm currently assuming that F12out = Ftotal out. CHANGE.

F_caco3_Start = F_12_in − F_C12_red_start; %this is actually F12
caco3 start

k_caco3 = F_caco3_Start./((dic.*1029.*VolOcean)./(1+
(((((del_C_foram_start + d13C_offset)./1000)+1).*stdratio_C)));     %
first order rate constant for CaCO3 burial in /yr. calculated such
that carbon starts in steady state

Fpy_start = F_12_in.*(((del_C_for_py_start + d13C_offset)−
del_in_C)./−eps_C);

Fredarray = zeros(1,length(timearray));
Fredarray(1,1:rampstart./timestep) = F_C12_red_start;
Fredarray (1,rampstart./timestep +1 :rampend./timestep) =
F_12_in.*((del_C_foram_ramp + d13C_offset)−del_in_C)./(−eps_C);
Fredarray(1,rampend./timestep+1:end) = F_12_in.*((del_C_foram_end +
d13C_offset)−del_in_C)./(−eps_C); %*******

Fpyarray = zeros(1,length(timearray));
Fpyarray(1,1:rampstart./timestep) = F_12_in.*(((del_C_for_py_start +
d13C_offset)−del_in_C)./−eps_C);
Fpyarray (1,rampstart./timestep +1 :rampend./timestep) =
F_12_in.*((del_C_for_py_ramp + d13C_offset)−del_in_C)./(−eps_C);
Fpyarray(1,rampend./timestep+1:end) = F_12_in.*((del_C_for_py_end +
d13C_offset)−del_in_C)./(−eps_C);

R_start=xlsread('d13C input data.xlsx',2,'X17'); %This is "f_marine"
R_ramp =xlsread('d13C input data.xlsx',2,'X18:X151');
R_end=1;
R_rampstart= 1e7;
R_rampend =2.34e7;
%R_jumptime = 2.34e7;
R_timestep =0.1e6;
R_runtime = 76e6;              % run length in years
R_timearray = 0:R_timestep:R_runtime;     % time arrary to pair with
F_in array
R_array = zeros(1,length(R_timearray));
R_array(1,1:R_rampstart./R_timestep) = R_start;
R_array (1,R_rampstart./R_timestep +1 :R_rampend./timestep) =
R_ramp;
R_array(1, R_rampend./R_timestep+1:end) = R_end;


CS_0 = (32./12).*0.860028185;
CS_end = (32./12).*0.860028185;
jumptime = 2.34e7;               % time of step change in F in
```

```matlab
CS_timestep = 1e5;
CS_runtime = 76e6;                % run length in years
CS_timearray = 0:CS_timestep:CS_runtime;    % time arrary to pair
with F_in array
CS_array = zeros(1,length(CS_timearray));
CS_array(1,1:jumptime./CS_timestep) = CS_0;
CS_array(1, jumptime./CS_timestep+1:end) = CS_end;
%
%
%
eps_S_mar = -35.5;
eps_S_ter = -3;
eps_ave = R_start*eps_S_mar + (1-R_start)*eps_S_ter;
% eps_0 =   -32;
% eps_new =   -37;
% eps_jumptime = 2.34e7;
% eps_timestep = 1e5;
% eps_runtime = 700e6;              % run length in years
% eps_timearray = 0:eps_timestep:eps_runtime;    % time arrary to
pair with F_in array
% eps_array = zeros(1,length(eps_timearray));
% eps_array(1,1:eps_jumptime./eps_timestep) = eps_0;
% eps_array(1, eps_jumptime./eps_timestep+1:end) = eps_new;
```