

Contents

- [Calibration - Correction demo on tetR repression data](#)
- [Visualize the MCMC Chains and posterior distributions](#)
- [Visualize the resulting 'fits' of the model to the data](#)
- [The test circuit \(Correction Step\)](#)
- [Correction Demo Figure](#)
- [compute the % correction](#)

Calibration - Correction demo on tetR repression data

(c) Vipul Singhal, California Institute of Technology, 2018

```
% % clean up things
clear all
close all
clc

%
st = dbstack('-completenames');
fp = st(1).file;
slashes = regexp(fp, '/');
projdir = fp(1:slashes(end)-1);
addpath(genpath(projdir));
```

Visualize the MCMC Chains and posterior distributions

Setup an array of all the .mat files where the calibration data posterior distributions are stored.

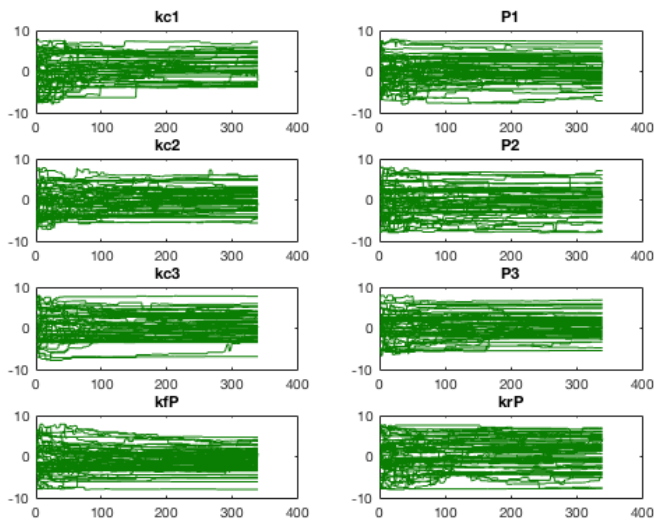
```
calib_SD = {'t015_calib_20171023_151627_1_MBP'
't015_calib_20171023_151627_2_MBP'
't015_calib_20171023_151627_3_MBP'
't015_calib_20171023_151627_4_MBP'
't015_calib_20171023_151627_5_MBP'
't015_calib_20171023_151627_6_MBP'
't015_calib_20171023_151627_7_MBP'
't015_calib_20171023_151627_8_MBP'
't015_calib_20171023_151627_9_MBP'
't015_calib_20171023_151627_10_MBP'
't015_calib_20171023_151627_11_MBP'
't015_calib_20171023_151627_12_MBP'
't015_calib_20171023_151627_13_MBP'
't015_calib_20171023_151627_14_MBP'
't015_calib_20171023_151627_15_MBP'
't015_calib_20171023_151627_16_MBP'
't015_calib_20171023_151627_17_MBP'
't015_calib_20171023_151627_18_MBP'
't015_calib_20171023_151627_19_MBP'
't015_calib_20171023_151627_20_MBP'};

% Number of walkers used in the ensemble MCMC:
nW = 600;

% parameter legends for plotting parameter posterior distributions later
legends = {'kc1' 'P1' 'kc2' 'P2' 'kc3' 'P3', 'kfp', 'krp'};

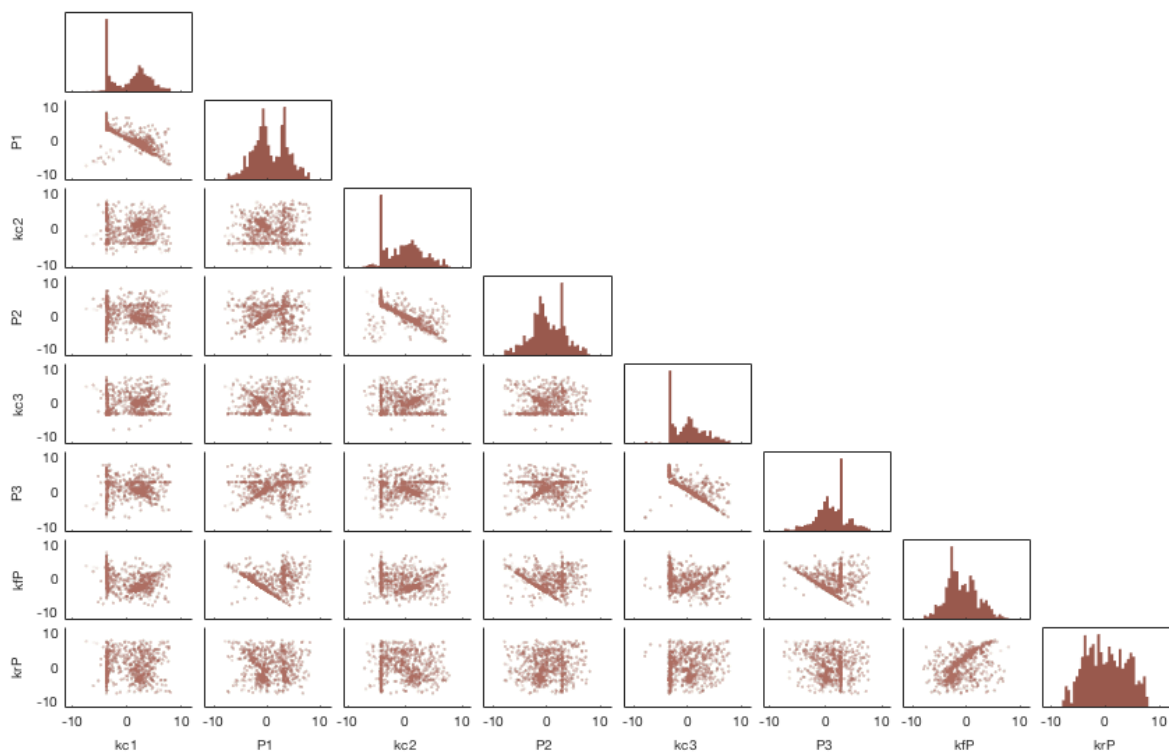
% Concatenate the parameter arrays drawn from the .mat files.
mcat = catMC(calib_SD);

% Plot the MCMC chains (for 1/10 of the walkers for easy visualization)
plotChains(mcat(:,1:10:end,:), nW, legends );
```



Plot pairwise projections of the joint posterior distribution.

```
figure
ecornerplot_vse(mcat(:,:(end-20):end), 'scatter', true, 'transparency', 0.025, ...
    'color', [.6 .35 .3], 'names', legends);
```



Visualize the resulting 'fits' of the model to the data

Overall idea: Pick 500 points from the parameter posterior distribution, generate trajectories, then take means, medians, standard deviations etc.

```
load('t015_calib_20171023_151627_5_MBP', 'm')
mstacked = m(:, :)';
nptstotal = size(mstacked, 1);
npts = 500;

medians_calib = median(mstacked);

kc_calib3 = medians_calib(5); % == -0.28209
P_calib3 = medians_calib(6); % == 1.3714
```

```
% Arbitrary points from the calibration dataset to generate the
% calibration fit figures.

mcat_conv = mcat(:, :, end-10:end);
mstacked_conv = mcat_conv(:, :);
paramid = randperm(size(mstacked_conv, 1), npts);

params_to_use_calib = mstacked_conv(paramid, :);

% use arbitrary point:
calib_arbitrary_point = mstacked(5586, :);
```

Get the points in the calibration step parameter space that all fit the median values k_c _calib3 and P _calib3. The ESP values for the -

```
tol = 0.044; % 0.04377298... This tol was found with manual bisection
% search type tweaking.

p_indices = ...
    intersect(...
        find(mstacked(:, 6) > P_calib3-tol),...
        find(mstacked(:, 6) < P_calib3+tol));

kc_indices = ...
    intersect(...
        find(mstacked(:, 5) > kc_calib3-tol),...
        find(mstacked(:, 5) < kc_calib3+tol));

valid_indices = intersect(p_indices, kc_indices);

% These are the points corresponding to the P_calib3 and kc_calib3
common_calib = mstacked(valid_indices(1), :);
```

```
envname = {'VS', 'MP', 'SG'};
% initialize things for the simulation and plotting.
load('t015_calib_20171023_151627_20_MBP', 'tvec', 'nW', 'model_calib',...
    'dosevals_calib', 'dosemap_calib', 'calibration_data', 'pmap_calib',...
    'nSp_calib', 'idMS_calib' )

nMS = size(calibration_data, 2); % nMS = 1 here, since only GFP is measured
nICs = size(dosevals_calib, 2); % ICs, GFP DNA = [1 2 5 10 20] (nM)

nEnv = size(calibration_data, 4);

espIX = pmap_calib{1};
esspIX = pmap_calib{2};
cspIX = pmap_calib{3};

nESP = length(espIX); % the ESP indices in the model (not in logpjoint)
nESSP = length(esspIX); % the Env specific species indices in the model (not in logpjoint)
nCSP = length(cspIX); % the CSP indices in the model (not in logpjoint)

icvec = zeros(nSp_calib, 1);

simulatedtraj = zeros(length(tvec(1:13)), nMS, nICs, npts, nEnv);
maxGFP_sd = 0;
```

simulate the calibration model for all the randomly picked points from the posterior distribution

```
for kk = 1:npts
    logpjoint = params_to_use_calib(kk, :);
    cspindices = ((nESSP + nESP)*nEnv+1):length(logpjoint);
    paramvec = zeros(nESP+nCSP, 1);
    logpcsp = logpjoint(cspindices);
    paramvec(cspIX) = logpcsp;

    for envid = 1:nEnv
        espindices = (envid-1)*(nESP+nESSP) + (1:nESP);
        logpesp = logpjoint(espindices);
        paramvec(espIX) = logpesp;
        esspindices = ((envid-1)*(nESP+nESSP) + nESP + 1):envid*(nESP+nESSP);

        % set the values of the initial condition vector to the parameters
        icvec(esspIX) = exp(logpjoint(esspindices));

        for doseID = 1:nICs
            icvec(dosemap_calib) = dosevals_calib(:, doseID);

            % simulate the model
            [~, simudata] = model_calib(paramvec, icvec, tvec(1:13));
            for msid = 1:nMS
```

```

simulatedtraj(:,msid, doseID, kk, envid) = simuldata(:, idMS_calib(msid));
end
end
end
end

```

Compute the means and standard deviations

```

meanvals = mean(simulatedtraj, 4);
sdvals= std(simulatedtraj,0, 4);
maxvals = squeeze(max(max(max(meanvals+sdvals,[], 1), [], 3), [], 5)); % 1 by nMS array.
lineStyles = linspace(nICs,'sequential');
hd = zeros(nICs, 1); % data trajectory handles
hm = zeros(nICs, 1); % model fit mean trajectory handles
hsd = zeros(nICs, 1); % model fit sd trajectory handles (patch objects)

```

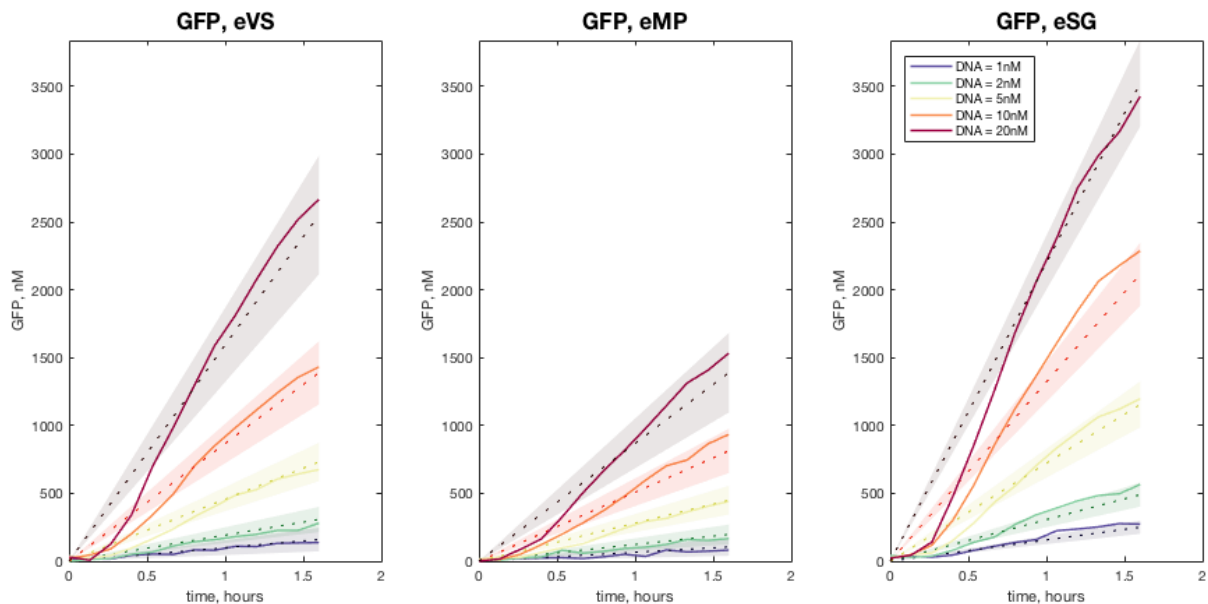
```

hd = zeros(nICs, 1); % data trajectory handles
hm = zeros(nICs, 1); % model fit mean trajectory handles
hsd = zeros(nICs, 1); % model fit sd trajectory handles (patch objects)

for msid = 1:nMS
    figure
    ss = get(0, 'screensize');
    set(gcf, 'Position', [50 100 ss(3)/1.6 ss(4)/2.3]);

    for j= 1:nEnv
        subplot(1, nEnv,j);
        for i = 1:nICs
            linearidx = nEnv*(i-1)+j;
            % each index corresponds to a dose environment pair. (ie, each
            % line and patch have a common index)
            hd(i)=plot(tvec(1:13)/3600,1000*calibration_data(1:13,msid, i,j ),...
                'color',lineStyles(i, :), 'linewidth',1.4);
            hold on
            [hm(i), hsd(i)] = boundedline(tvec(1:13)/3600,...
                meanvals(:, msid, i, 1, j), sdvals(:, msid, i, 1, j));
            set(hsd(i), 'FaceColor', lineStyles(i, :).^4, 'FaceAlpha', 0.1);
            set(hm(i), 'Color', lineStyles(i, :).^4, 'LineStyle', ':');
            hold on
            set(hm(i), 'LineWidth', 0.8)
        end
        set(gca, 'Ylim', [0, round(maxvals(msid))])
        title(sprintf('GFP, e%s', envname{j}), 'FontSize', 16)
        xlabel('time, hours')
        ylabel('GFP, nM')
    end
    legend(hd, {'DNA = 1nM', 'DNA = 2nM', 'DNA = 5nM', 'DNA = 10nM', 'DNA = 20nM'},...
        'Location', 'NorthWest')
end

```



The test circuit (Correction Step)

We start with a circuit description tetR_repression: tet repression model, single step, first 3 hours.

$D_T + P - D_{T:P} \rightarrow D_T + P + T$

$D_G + P - D_{G:P} \rightarrow D_G + P + G$

$2T - T_2$

$D_G + T_2 - D_{G:T_2}$

Next we visualize the Markov chains and posterior distributions for the test circuit.

```
corr_SD = { 't015_corr1_20171023_151627_2_MBP'
't015_corr1_20171023_151627_3_MBP'
't015_corr1_20171023_151627_4_MBP'
't015_corr1_20171023_151627_5_MBP'
't015_corr1_20171023_151627_6_MBP'
't015_corr1_20171023_151627_7_MBP'
't015_corr1_20171023_151627_8_MBP'
't015_corr1_20171023_151627_9_MBP'
't015_corr1_20171023_151627_10_MBP'
't015_corr1_20171023_151627_11_MBP'
't015_corr1_20171023_151627_12_MBP'
't015_corr1_20171023_151627_13_MBP'
't015_corr1_20171023_151627_14_MBP'
't015_corr1_20171023_151627_15_MBP'
't015_corr1_20171023_151627_16_MBP'
't015_corr1_20171023_151627_17_MBP'};

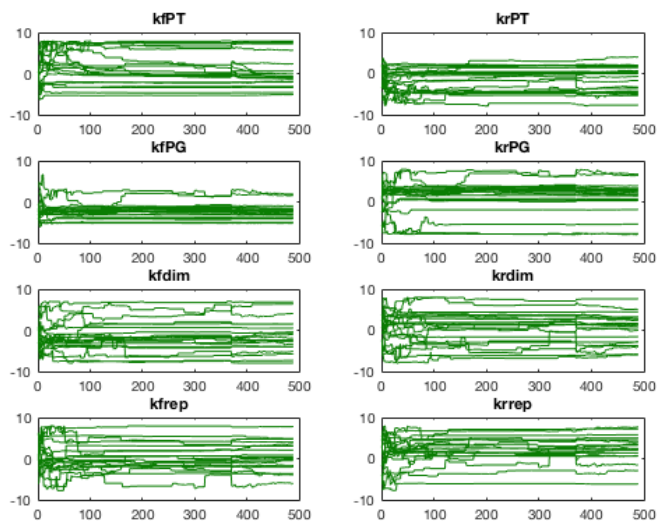
corr_SD2 = { 't015_corr1_20171023_151627_11_MBP'
't015_corr1_20171023_151627_12_MBP'
't015_corr1_20171023_151627_13_MBP'
't015_corr1_20171023_151627_14_MBP'
't015_corr1_20171023_151627_15_MBP'
't015_corr1_20171023_151627_16_MBP'
't015_corr1_20171023_151627_17_MBP'};

nW = 600;
legends = {'kfPT' 'krPT' 'kfPG' 'krPG' 'kfdim' 'krdim' 'kfrep' 'krrep'};

mcat = catMC(corr_SD);
```

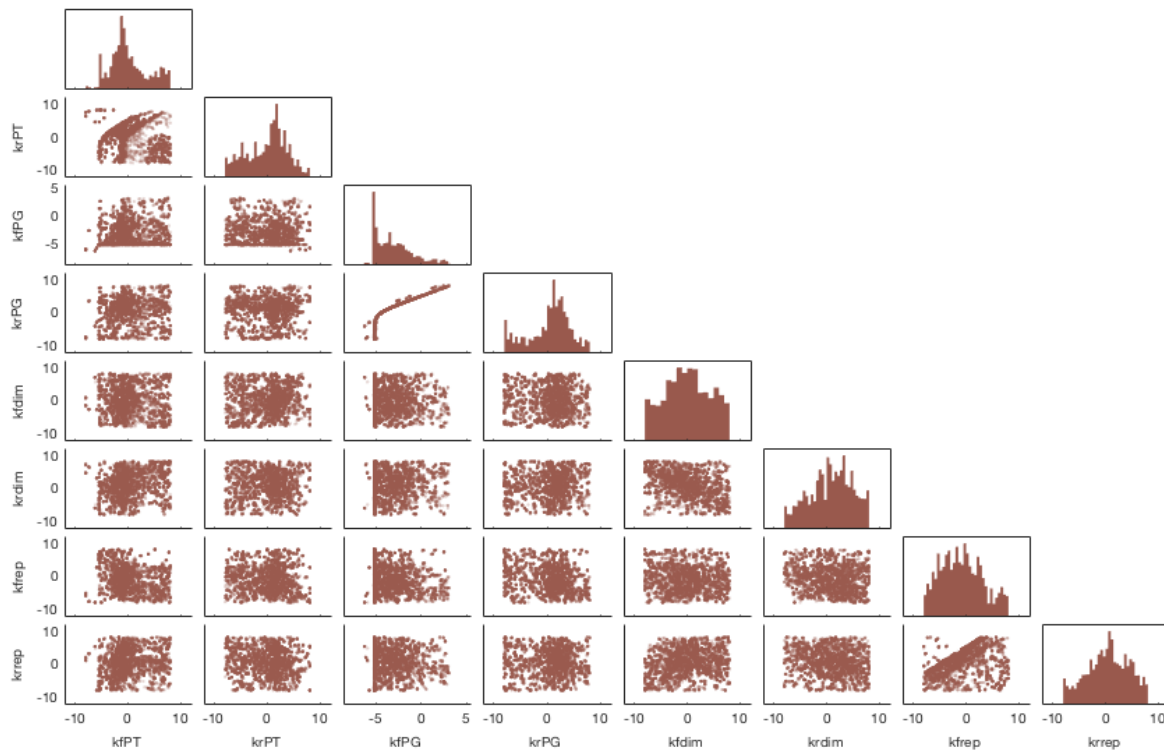
Plot the chains for all the 17 iterations, for 20 of the walkers (for easier visualization).

```
plotChains(mcat(:,1:30:end, :), nW, legends );
```



Plot the scatterplot for the last 7 iterations

```
mcat_converged= catMC(corr_SD2);
figure
ecornerplot_vse(mcat_converged(:, :, 1:10:end),...
'scatter', true,'transparency',0.25, 'color',[.6 .35 .3], ...
'names', legends);
```



Correction Demo Figure

Next we create the correction demo figure. This figure is arranged into 3 columns and nICs number of rows. Each row corresponds to one dose (initial condition). Within each row, the subplot corresponding to the first column has the test circuit behavior in the two environments of interest, the candidate environment eSG and the reference environment (eVS). The second column has the same two trajectories, but in addition has the model fit to the candidate environment data. The third column has the 'corrected' behavior, along with the two data trajectories.

```
envrefID = 1; % can be changed to 2 to generate the correction from 3 to 2.
envcandID = 3;

m_rearranged = mcat_converged(:, :);
nptstotal = size(m_rearranged, 1);

paramid = randperm(nptstotal, npts);
params_to_use_corr = m_rearranged(paramid, :);
envname = {'VS', 'MP', 'SG'};
load('t015_corr1_20171023_151627_11_MBP', 'tvec', 'nW', 'model_corr', ...
     'dosevals_corr', 'dosemap_corr', 'correction_data', 'pmap_corr', ...
     'nSp_corr', 'idMS_corr');

nMS = size(correction_data, 2); % nMS = 1 here, since only GFP is measured
nICs = size(dosevals_corr, 2); % ICs, tetR DNA = [0 0.25 0.5 0.75 1 2 5 10] (nM)

nEnv_total = size(correction_data, 4);
% the total number of environments for which we have data.
nEnv_used = 2;
% the number of environments considered to demonstrate the correction procedure.
nEnv_estimated = 1;
% the number of environments on which the parameter estimation was performed.

espIX = pmap_corr{1};
esspIX = pmap_corr{2};
cspIX = pmap_corr{3};

nESP = length(espIX); % the ESP indices in the model (not in logpjoint)
nESSP = length(esspIX); % the Env specific species indices in the model (not in logpjoint)
nCSP = length(cspIX); % the CSP indices in the model (not in logpjoint)

icvec = zeros(nSp_corr, 1);
```

we will have 2 sets of simulated trajectories, one for the candidate env and one for the reference. Therefore, the number of environments used is nEnv_used (= 2).

```
simulatedtraj_corrstep1 = zeros(length(tvec(1:13)), nMS, nICs, npts);
simulatedtraj_corrstep2 = zeros(length(tvec(1:13)), nMS, nICs, npts);

simtraj_cs2_cspfix = zeros(length(tvec(1:13)), nMS, nICs, npts);
```

simulate the correction model for all the randomly picked points from the posterior distribution, fixing the ESPs and ESSPs to the candidate environments values.

```
for kk = 1:npts
    logpjoint_corr1 = [-0.2821 1.3714 params_to_use_corr(kk, :)];
    cspindices = ((nESSP + nESP)*nEnv_estimated+1):length(logpjoint_corr1);
    paramvec = zeros(nESP+nCSP, 1);
    logpcsp = logpjoint_corr1(cspindices);
    paramvec(cspIX) = logpcsp;
    logpesp = logpjoint_corr1(1:nESP);
    paramvec(espIX) = logpesp;
    esspindices = (nESP + 1):(nESP+nESSP);

    % set the values of the initial condition vector to the parameters
    icvec(esspIX) = exp(logpjoint_corr1(esspindices));

    for doseID = 1:nICs
        icvec(dosemap_corr) = dosevals_corr(:, doseID);
        % simulate the model
        [~, simudata] = model_corr(paramvec, icvec, tvec(1:13));
        for msid = 1:nMS
            simulatedtraj_corrstep1(:,msid, doseID, kk) = simudata(:, idMS_corr(msid));
        end
    end
end
```

Compute the mean and standard deviations for correction step 1

```
meanvals_corrstep1 = mean(simulatedtraj_corrstep1, 4);
sdvals_corrstep1 = std(simulatedtraj_corrstep1, 0, 4);
maxvals_corrstep1 = squeeze(max(max(max(meanvals_corrstep1+sdvals_corrstep1,...
    [], 1), [], 3), [], 5)); % 1 by nMS array.
```

Also, simulate the trajectories in the reference environment. here we randomly mix and match points from the reference environments environment specific parameters and species, and the CSP from correction step 1.

```
refsID = ((envrefID-1)*(nESSP + nESP)+1); % reference ESP start ID
refeID = (envrefID*(nESSP + nESP)); % reference ESP end ID

refmedians = calib_arbitrary_point(:, refsID:refeID);
ref_ESPs_sharedCSPs = common_calib(refsID:refeID);
```

other option: params_to_use_calib(kk,... ((envrefID-1)*(nESSP + nESP)+1):(envrefID*(nESSP + nESP))), though this is a bit buggy right now

```
for kk = 1:npts
    logpjoint_corrstep2 = [refmedians params_to_use_corr(kk, :)];
    cspindices = ((nESSP + nESP)+1):length(logpjoint_corrstep2);
    paramvec = zeros(nESP+nCSP, 1);
    logpcsp = logpjoint_corrstep2(cspindices);
    paramvec(cspIX) = logpcsp;
    logpesp = logpjoint_corrstep2(1:nESP);
    paramvec(espIX) = logpesp;
    esspindices = (nESP + 1):(nESP+nESSP);

    % set the values of the initial condition vector to the parameters
    icvec(esspIX) = exp(logpjoint_corrstep2(esspindices));

    for doseID = 1:nICs
        icvec(dosemap_corr) = dosevals_corr(:, doseID);

        % simulate the model
        [~, simudata] = model_corr(paramvec, icvec, tvec(1:13));
        for msid = 1:nMS
            simulatedtraj_corrstep2(:,msid, doseID, kk) = ...
                simudata(:, idMS_corr(msid));
        end
    end
end

% Compute the mean and standard deviations for correction step 1
meanvals_corrstep2 = mean(simulatedtraj_corrstep2, 4);
sdvals_corrstep2 = std(simulatedtraj_corrstep2, 0, 4);
maxvals_corrstep2 = squeeze(max(max(max(meanvals_corrstep2+sdvals_corrstep2,...
    [], 1), [], 3), [], 5)); % 1 by nMS array.
```

other option: params_to_use_calib(kk,... ((envrefID-1)*(nESSP + nESP)+1):(envrefID*(nESSP + nESP))), though this is a bit buggy right now

```
for kk = 1:npts
    logpjoint_corrstep2_cspfix = [ref_ESPs_sharedCSPs params_to_use_corr(kk, :)];
    cspindices = ((nESSP + nESP)+1):length(logpjoint_corrstep2_cspfix);
```

```

paramvec = zeros(nESP+nCSP, 1);
logpcsp = logpjoint_corrstep2_cspfix(cspindices);
paramvec(cspIX) = logpcsp;
logpesp = logpjoint_corrstep2_cspfix(1:nESP);
paramvec(espIX) = logpesp;
esspindices = (nESP + 1):(nESP+nESSP);

% set the values of the initial condition vector to the parameters
icvec(esspIX) = exp(logpjoint_corrstep2_cspfix(esspindices));

for doseID = 1:nICs
    icvec(dosemap_corr) = dosevals_corr(:, doseID);

    % simulate the model
    [~, simudata] = model_corr(paramvec, icvec, tvec(1:13));
    for msid = 1:nMS
        simtraj_cs2_cspfix(:,msid, doseID, kk) =...
            simudata(:, idMS_corr(msid));
    end
end

% Compute the mean and standard deviations for correction step 1
meanvals_corrstep2_cs2_cspfix = mean(simtraj_cs2_cspfix, 4);
sdvals_corrstep2_cs2_cspfix= std(simtraj_cs2_cspfix,0, 4);
maxvals_corrstep2_cs2_cspfix = squeeze(max(max(max(meanvals_corrstep2_cs2_cspfix+...
    sdvals_corrstep2_cs2_cspfix,...
    [], 1), [], 3), [], 5)); % 1 by nMS array.

% compute the max of the axis jointly for corrstep 1 and 2.
maxvals_corr = max([maxvals_corrstep1; maxvals_corrstep2;...
    maxvals_corrstep2_cs2_cspfix], [], 1);

```

Initalize arrays for handles to the graphics objects.

```

lineStyles = linspace(2*nICs,'sequential');
hd_cand = zeros(nICs, 3); % data trajectory handles for candidate environment
hd_ref = zeros(nICs, 3); % data trajectory handles for reference environment
hm_cand = zeros(nICs, 1); % model fit mean trajectory handles
hsd_cand = zeros(nICs, 1); % model fit sd trajectory handles (patch objects)
hm_ref = zeros(nICs, 1); % model prediction mean trajectory handles
hsd_ref = zeros(nICs, 1); % model prediction sd trajectory handles (patch objects)

```

create the 3 column subplot

```

nICs = 4;
for msid = 1:nMS
    maxvals_corr(msid) = 1500;
    figure
    ss = get(0, 'screensize');
    set(gcf, 'Position', [50 100 ss(3)/1.6 ss(4)/1.4]);

    % for each initial condition row
    for i = 1:nICs
        % column 1: just the experimental data
        linearidx = 4*(i-1)+1;
        subplot(nICs, 4,linearidx);

        hd_ref(i, 1)=plot(tvec(1:13)/3600,1000*correction_data(1:13,msid, i, envrefID),...
            'color',lineStyles(i, :) , 'linewidth',0.8);
        hold on
        hd_cand(i, 1)=plot(tvec(1:13)/3600,1000*correction_data(1:13,msid, i,envcandID ),...
            'color',lineStyles(nICs+i, :) , 'linewidth',0.8);
        hold on
        set(gca, 'Ylim', [0, round(maxvals_corr(msid))])
        set(gca, 'Xlim', [0, 1.6])
        title(sprintf('Experimental data, tetR DNA = %0.2g',...
            dosevals_corr(2, i)), 'FontSize', 12)
        xlabel('time, hours')
        ylabel('GFP, nM')
        legend([hd_ref(i, 1), hd_cand(i, 1)], ...
            {'Reference Extract', 'Candidate Extract'}, 'Location', 'NorthWest')

        % column 2: overlay correction step 1 fit (CSP estimation)
        linearidx = 4*(i-1)+2;
        subplot(nICs, 4,linearidx);
        hd_ref(i, 2)=plot(tvec(1:13)/3600,1000*correction_data(1:13,msid, i, envrefID),...
            'color',lineStyles(i, :) , 'linewidth',0.8);
        hold on
        hd_cand(i, 2)=plot(tvec(1:13)/3600,1000*correction_data(1:13,msid, i,envcandID ),...
            'color',lineStyles(nICs+i, :) , 'linewidth',0.8);
        hold on
        [hm_cand(i), hsd_cand(i)] = boundedline(tvec(1:13)/3600,...
            meanvals_corrstep1(:, msid, i, 1), sdvals_corrstep1(:, msid, i, 1));
    end
end

```



```

set(hsd_cand(i), 'FaceColor', lineStyles(nICs+i, :).^4, 'FaceAlpha', 0.1);
set(hm_cand(i), 'Color', lineStyles(nICs+i, :).^4, 'LineStyle', ':');
hold on
set(hm_cand(i), 'LineWidth', 1)
set(gca, 'Ylim', [0, round(maxvals_corr(msid))])
set(gca, 'Xlim', [0, 1.6])
title(sprintf('Correction Step 1, tetR DNA = %0.2g',...
    dosevals_corr(2, i)), 'FontSize', 12)
xlabel('time, hours')
legend([hd_ref(i, 2), hd_cand(i, 2), hm_cand(i)], ...
    {'Reference Extract', 'Candidate Extract', 'Model Fit (mean, sd)'},...
    'Location', 'NorthWest')
%
    ylabel('GFP, nM')

% column 3: overlay correction step 2 prediction instead of
% correction step 1 fit. ("corrected behavior")
linearidx = 4*(i-1)+3;
subplot(nICs, 4, linearidx);
hd_ref(i, 3)=plot(tvec(1:13)/3600,1000*correction_data(1:13,msid, i, envrefID),...
    'color',lineStyles(i, :), 'linewidth',0.8);
hold on
hd_cand(i, 3)=plot(tvec(1:13)/3600,1000*correction_data(1:13,msid, i,envcandID ),...
    'color',lineStyles(nICs+i, :), 'linewidth',0.8);
hold on
[hm_ref1(i), hsd_ref1(i)] = boundedline(tvec(1:13)/3600,...
    meanvals_corrstep2(:, msid, i, 1), sdvals_corrstep2(:, msid, i, 1));
set(hsd_ref1(i), 'FaceColor', lineStyles(i, :).^4, 'FaceAlpha', 0.1);
set(hm_ref1(i), 'Color', lineStyles(i, :).^4, 'LineStyle', ':');
hold on
set(hm_ref1(i), 'LineWidth', 1)
set(gca, 'Ylim', [0, round(maxvals_corr(msid))])
set(gca, 'Xlim', [0, 1.6])
title(sprintf('Correction Step 2, tetR DNA = %0.2g',...
    dosevals_corr(2, i)), 'FontSize', 12)
xlabel('time, hours')
legend([hd_ref(i, 3), hd_cand(i, 3), hm_ref1(i)],...
    {'Reference Extract', 'Candidate Extract', ...
    ''Corrected'' Trajectories (mean, sd)'}, 'Location', 'NorthWest')
%
    ylabel('GFP, nM')

% column 4: overlay correction step 2 prediction instead of
% correction step 1 fit. ("corrected behavior") WITH CSP FIXING
linearidx = 4*(i-1)+4;
subplot(nICs, 4, linearidx);

% reference experimental data
hd_ref(i, 4)=plot(tvec(1:13)/3600,1000*correction_data(1:13,msid, i, envrefID),...
    'color',lineStyles(i, :), 'linewidth',0.8);
hold on

% candidate experimental data
hd_cand(i, 4)=plot(tvec(1:13)/3600,1000*correction_data(1:13,msid, i,envcandID ),...
    'color',lineStyles(nICs+i, :), 'linewidth',0.8);
hold on

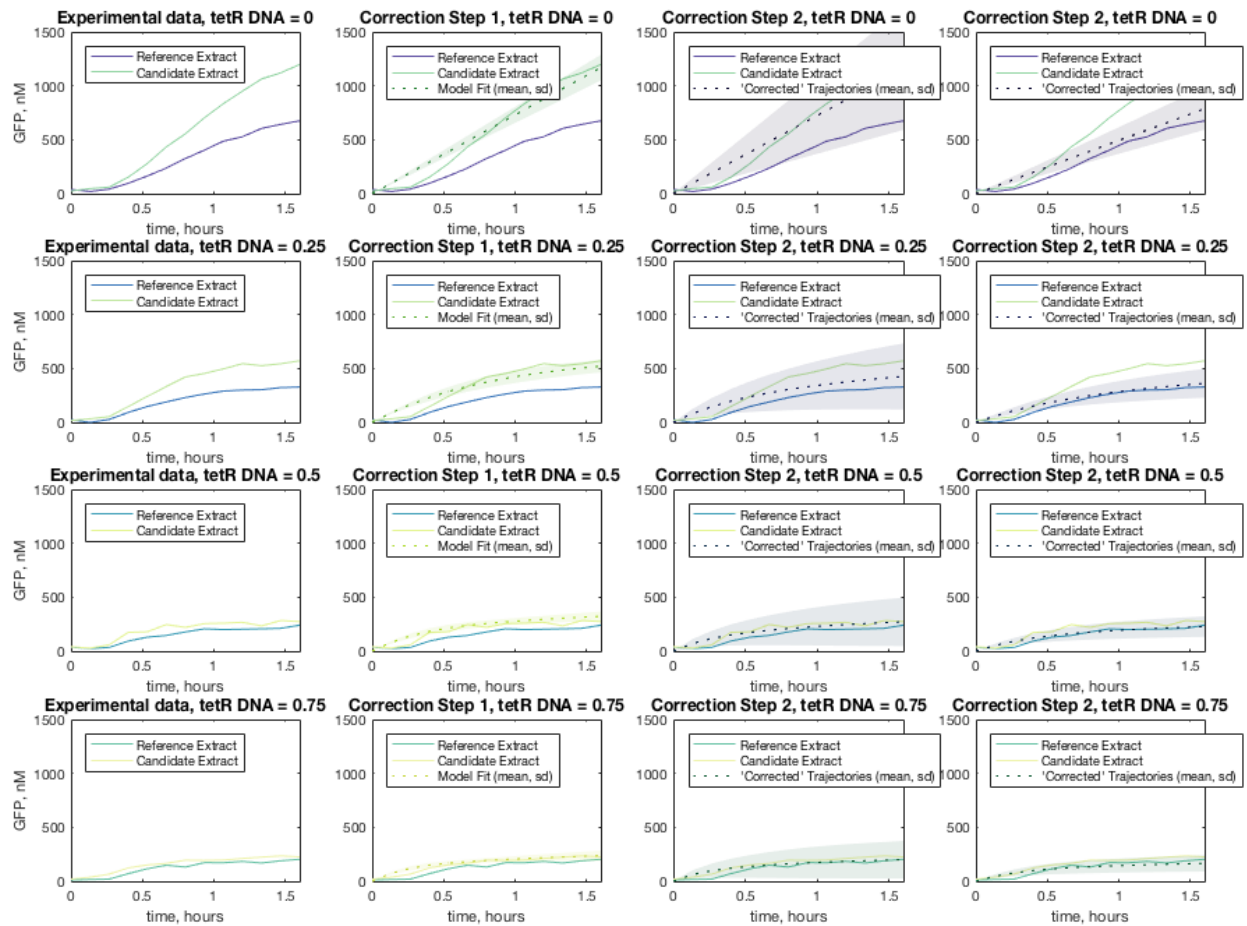
% plot the corrected traj mean and standard deviation (as a matlab
% patch class object)
[hm_ref2(i), hsd_ref2(i)] = boundedline(tvec(1:13)/3600,...
    meanvals_corrstep2_cs2_cspfix(:, msid, i, 1), ...
    sdvals_corrstep2_cs2_cspfix(:, msid, i, 1));
set(hsd_ref2(i), 'FaceColor', lineStyles(i, :).^4, 'FaceAlpha', 0.1);
set(hm_ref2(i), 'Color', lineStyles(i, :).^4, 'LineStyle', ':');
hold on

set(hm_ref2(i), 'LineWidth', 1)
set(gca, 'Ylim', [0, round(maxvals_corr(msid))])
set(gca, 'Xlim', [0, 1.6])

title(sprintf('Correction Step 2, tetR DNA = %0.2g',...
    dosevals_corr(2, i)), 'FontSize', 12)
xlabel('time, hours')
legend([hd_ref(i, 4), hd_cand(i, 4), hm_ref2(i)],...
    {'Reference Extract', 'Candidate Extract', ...
    ''Corrected'' Trajectories (mean, sd)'}, 'Location', 'NorthWest')

%
% Compute the \% correction
% normorig = norm(1000*correction_data(1:13,msid, i,...
%     envrefID) - 1000*correction_data(1:13,msid, i,envcandID ))
% normreduced = norm(1000*correction_data(1:13,msid, i,...
%     envrefID) - meanvals_corrstep2(:, msid, i, 1))
end
end

```



compute the % correction

```

nICs = 4;
normorig = zeros(nICs, 1);
normreduced = zeros(nICs, 1);
alpha = zeros(nICs, 1);
normreduced_cspfix = zeros(nICs, 1);
alpha_cspfix = zeros(nICs, 1);
for msid = 1:nMS
% for each initial condition row
    for i = 1:nICs
        % distance between the reference and candidate experimental data.
        normorig(i) = norm(1000*correction_data(1:13,msid, i,...
            envrefID) - 1000*correction_data(1:13,msid, i,envcandID ));

        % distance between the mean of the
        normreduced(i) = norm(1000*correction_data(1:13,msid, i, envrefID)...
            - meanvals_corrstep2(:, msid, i, 1));

        normreduced_cspfix(i) = norm(1000*correction_data(1:13,msid, i, envrefID)...
            - meanvals_corrstep2_cs2_cspfix(:, msid, i, 1));

        alpha(i) = ...
            max(sdvls_corrstep2(:, 1, i, 1))/max(sdvls_corrstep1(:, 1, i, 1));

        alpha_cspfix(i) = ...
            max(sdvls_corrstep2_cs2_cspfix(:, 1, i, 1))/max(sdvls_corrstep1(:, 1, i, 1));
        % One idea of defining the metric could be to weight the correction
        % by the standard deviation somehow. see the notability note:
        % Note Apr 1, 2018 April-Sep logbook --- aug 10 entry.

        % nrm2(i) = ((normorig(i)-normreduced(i))/(normorig(i)));

        % % infy norm (want the max difference to come down by 2X
        % normorig(i) = norm(1000*correction_data(1:13,msid, i, envrefID)...
        % - 1000*correction_data(1:13,msid, i,envcandID ), Inf);
        % normreduced(i) = norm(1000*correction_data(1:13,msid, i, envrefID)...
        % - meanvals_corrstep2(:, msid, i, 1), Inf);
        % nrmi(i) = ((normorig(i))/(normreduced(i)));

```

```

%      nrm2i(i) = ((normorigi(i)-normreducedi(i))/(normorigi(i)));
%
      end
end

RR11 = sum(normreduced)/sum(normorig);
RR12 = sum(alpha.*normreduced)/sum(normorig);
RR13 = sum(normreduced_cspfix)/sum(normorig);
RR14 = sum(alpha_cspfix.*normreduced_cspfix)/sum(normorig);

RR21 = sum(normreduced./normorig)/nICs;
RR22 = sum(alpha.*normreduced./normorig)/nICs;
RR23 = sum(normreduced_cspfix./normorig)/nICs;
RR24 = sum(alpha_cspfix.*normreduced_cspfix./normorig)/nICs;

metricvals = [RR11 RR12 RR13 RR14 RR21 RR22 RR23 RR24]

```

metricvals =

Columns 1 through 7

0.7906 3.8188 0.2971 0.5541 0.7807 3.6357 0.4207

Column 8

0.7786

Published with MATLAB® R2017b