Documentation on Recipe for Gates Data

Overall Codes Used:

TensorExplorationPARAFAC.m

YenerTensor.m

Adjust APGAR Scores: ensure they are between 0-10:

Taken From: YenerTensor.m, Section 1

Inputs: the columns for APGAR scores from the initial matrix

What it does: subtracts 20 from APGAR scores of at least 20

The MATLAB code is:

for k = 1:size(num,1) % to go through all the subjects

if num(k,19) >= 20 % column for APGAR score after 1 minute

num(k,19) = num(k,19)-20; % subtract by 20

end

if num(k,20) >= 20 % column for APGAR score after 5 minutes

num(k,20) = num(k,20)-20; % subtract by 20

end

end

Choose subjects with all five time points:

Taken From: YenerTensor.m, Sections 2 and 3

Inputs: the subject ID column from the original matrix

What it does: counts the number of subject occurrences, only chooses the subjects with 5 five occurrences

The MATLAB code is:

%% Uniqueness

U = unique(num(:,2)); % get unique subjids

count = histc(num(:,2),U); % count the number of occurrences

%% Usable Subjects

subj = U(count==5); % take only subjects with all five time points

ib = find(ismember(num(:,2),subj));

final = num(ib,:); % only subjects with all five time points

Growth Data:

Weights, Heights

Taken From: YenerTensor.m, Section 4

Inputs: the height or weight column from the original matrix

What it does: finds how many values are missing:

If that is one, then fits a linear and quadratic regression on the remaining data. The fit with the higher R2 is used.

If that is two, then fits a linear regression on the remaining data.

If that is three, then the subject is removed.

The MATLAB code is: (also includes BMI)

%% Regression - Missing Points

for p = 1:size(subj,1) % interate through the total number of subjects

k = subj(p); % what subject number?

ib = find(ismember(final(:,2),k)); % dealing with one subject

heights = final(ib,5); % heights for that subject

bmis = final(ib,6); % bmis for that subject

weights = final(ib,4); % weights for that subject

times = final(ib,3); %times for that subject

where = isnan(final(ib,5)); % which values are missing?

count = sum(where); % how many are missing?

if count == 1 % only 1 missing time point

p1 = polyfit(times(~where),heights(~where),1); % linear fit

p2 = polyfit(times(~where),heights(~where),2); % quad fit

y1 = polyval(p1,times(~where));

r1 = rsq(heights(~where),y1); % linear r^2?

y2 = polyval(p2,times(~where));

r2 = rsq(heights(~where),y2); % quad r^2?

if r1>=r2 % if linear better than quad

heights(where) = polyval(p1,times(where));

bmis(where) = weights(where)./((heights(where)/100).^2);

else % if quad better than linear

heights(where) = polyval(p2,times(where));

bmis(where) = weights(where)./((heights(where)/100).^2);

end

final(ib,5) = heights;

final(ib,6) = bmis;

elseif count == 2 % 2 missing time points - linear fit

p1 = polyfit(times(~where),heights(~where),1);

miss = find(where == 1); % where are the missing values

heights(miss(1)) = polyval(p1,times(miss(1)));

heights(miss(2)) = polyval(p1,times(miss(2)));

bmis(where) = weights(where)./((heights(where)/100).^2);

final(ib,5) = heights;

final(ib,6) = bmis;

else % get rid of subjects >2 missing time points

final(ib,:) = [];

end

end

BMI:

Taken From: YenerTensor.m, Section 4

Input: the fitted and actual values from the height, weight imputation

What it does: calculates BMI from the values for each subject using the formula:



The MATLAB code is: see Heights, Weights for code.

BAZ, WAZ HAZ

Taken From: No overall code, done separately after Section 4 of YenerTensor.m

Input: the fitted and actual values from the height, weight, bmi imputation

What it does: calculates BAZ, HAZ, and WAZ from the imputed values based on WHO standards

The code is: Macros to calculate this using SAS, R, or STATA can be found here: http://www.who.int/childgrowth/software/en/

Characteristic Data:

Taken From: YenerTensor.m, Section 8

Input: all characteristic, height, weight data for subjects with all time points, from the original matrix along with the imputed height and weight data

What it does: gets either the mean or mode (depending on the type of characteristic data) based on the 10 most similar subjects based on height and weight, using this metric:



The MATLAB code is:

placeholder = sum(isnan(chara)); % where all the missing values are

% missing\_ids = zeros(size(chara,1),1); % which subjects had missing values

for k = 1:size(chara,2) % going through all the columns

if placeholder(k) > 0 % does that column have missing values?

C = chara(isnan(chara(:,k)),1);

% C = setdiff(ids, missing\_ids(:,1));

for c = 1:size(C,1)

% p = p + 1;

% missing\_ids(p,1) = C(c);

gender = chara(chara(:,1) == C(c),2); % gender of missing value

weights = transpose(abs(num(num(~ismember(num(:,2),C),3)==1,4)...

- num(num(:,2)==C(c) & num(:,3)==1,4))); % weight comparison

heights = transpose(abs(num(num(~ismember(num(:,2),C),3)==1,5)...

- num(num(:,2)==C(c) & num(:,3)==1,5))); % height comparison

together = [weights + heights; chara(~ismember(chara(:,1),C),2)';...

chara(~ismember(chara(:,1),C),k)']; % comparison; gender; value

[sorted,idx] = sort(together(1,:)); % sort by comparison value

together = together(:,idx); % organize them by comparison value

idx = find(together(2,:) == gender,10,'first'); % find the first 10 values

if ismember(k,[2 3 8 13])

chara(chara(:,1) == C(c),k) = mode(together(3,idx)); % take their mode

else

chara(chara(:,1) == C(c),k) = nanmean(together(3,idx)); % take their mean

end

end

end

end

Determine “significant” data:

Taken From: N/A

Input: tensor formulated from matrix

What it does: After you have run tensor through analysis, use discretion as to which columns of data to include

The MATLAB code is: N/A

Normalization

Taken From: TensorExplorationPARAFAC.m, Section 3

Input: tensor formulated from matrix

What it does: For growth data, it goes through each subject and subtracts the mean and standard deviation for each growth characteristic to normalize. For characteristic data, it goes through each column and subtracts by the mean and standard deviation of that column.

The MATLAB code is:

% normalization

for k = 8:size(gates\_tensor,2) % normalizing characteristics

for b = 1:size(gates\_tensor,3)

meaning = nanmean(gates\_tensor(:,k,1));

stdev = nanstd(gates\_tensor(:,k,1));

gates\_tensor(:,k,b) = (gates\_tensor(:,k,b) - meaning)/stdev;

end

end

for k = 1:size(gates\_tensor,1) % normalizing height, weight, bmi

for p = 1:3

avg = nanmean(gates\_tensor(k,p,:));

stdev = nanstd(gates\_tensor(k,p,:));

gates\_tensor(k,p,:) = (gates\_tensor(k,p,:)-avg)/stdev;

end

end

Flag SGA kids:

Taken From: TensorExplorationPARAFAC.m, Section 5

Input: unique subjects birth WAZ

What it does: gets the SGA status of each unique subject (this is calculated by using birth WAZ in the less than 10th percentile)

The MATLAB code is:

%% SGA Kids - Yener Tensor

% SGA kids

U = gates\_keep(:,1,1); % subjects

birthwaz = gates\_keep(:,5,1); % birth waz

SGA(:,1) = U; % subjects

SGA(:,2) = zeros(size(U)); % preallocating

SGA(birthwaz <=-2.33,2) = 1; % SGA kids

chara = [chara SGA(:,2)]; % add that characteristic

% gates\_SGA = gates\_keep(birthwaz <=-2.33,1:19,:);

Generate PARAFAC Model

Taken From: TensorExplorationPARAFAC.m, Section 7 (as part of scree plot and CORCONDIA plot) and Section 8 (as part of analysis)

Input: tensor formulated from matrix

What it does: creates a PARAFAC model for the data using the N-Way toolbox

The MATLAB code is:

SECTION 7:

%% Choosing the amount of PARAFAC components

% let's try centering?

[gates\_tensornew,means,scales]=nprocess(gates\_tensor,[0 0 0],[0 0 0]);

tensoring = gates\_tensornew;

err = zeros(1,3); % initializing the SSE

consistency = zeros(1,2); % initializing the consistency

for k = 1:3

[factors,~,err(k),~] = parafac(gates\_tensor,k);

consistency(k) = corcond(tensoring,factors);

end

% scree-plot

figure

plot(1:3,err);

title('Scree-Plot for PARAFAC model');

xlabel('Number of Components');

ylabel('SSE');

% 2 is best

% core consistency plot

figure

plot(1:3,consistency);

title('Core Consistency Plot');

xlabel('Number of Components');

ylabel('CONCORDIA');

% displays that 2 is best

SECTION 8:

%% PARAFAC

% create parafac model

[factors,it,err,corcondia,output] = parafac(tensoring,2,[0 0 0 0 10 0]);

% if options(3) = 2, MATLAB runs out of memory at the 3rd plot :(

[A,B,C] = fac2let(factors); % get loadings

Generate Scree Plot

Taken From: TensorExplorationPARAFAC.m, Section 7

Input: tensor formulated from matrix

What it does: creates a PARAFAC model for the data using the N-Way toolbox and calculates the sum of squared errors, then plots it for each number of model components versus the sum of squared errors

The MATLAB code is: see section 7 of code of creating a PARAFAC model

Generate CONCORDIA Number

Taken From: TensorExplorationPARAFAC.m, Section 7

Input: tensor formulated from matrix

What it does: creates a PARAFAC model for the data using the N-Way toolbox and calculates the CORCONDIA number, then plots it for each number of model components versus the CORCONDIA number

The MATLAB code is: see section 7 of code of creating a PARAFAC model

CONCORDIA Number?

Taken From: This decision sequence follows from the CONCORDIA Number section, TensorExplorationPARAFAC.m, Section 7

Input: CONCORDIA numbers and number of components of a PARAFAC model

What it does: decision sequence:

a. If the CORCONDIA number is high (close to 100%), then PARAFAC model is valid.

b. If the CORCONDIA number is mid-range (around 50%), the PARAFAC model should be reconsidered, perhaps with more constraints.

c. If the CORCONDIA number is low (close to 0%), the PARAFAC model is not valid.

The MATLAB code is: see section 7 of code of creating a PARAFAC model

Create a 3d plot for each time point for the original data, the modeled data, and the residuals

Taken From: TensorExplorationPARAFAC.m, Section 10

Input: tensor formulated from matrix

What it does: creates mesh plots for each time point based on the original tensor, the modeled data by PARAFAC, and the residual (original data – modeled data), respecively

The MATLAB code is:

%% Comparing Original to PARAFAC modeling

time\_points = [1 123 366 1462 2558];

% displaying original data

figure % plot

for k = 1:time\_pt % for each of the time points

subplot(3,2,k),mesh(tensoring(:,:,k));

title(sprintf('Time Point %g',time\_points(k)));

ylabel('Subject');

xlabel('Feature');

end

suptitle('Original Data');

% estimate model

[gates\_model] = nmodel(factors,[],0);

figure % plot model

for k = 1:time\_pt % for each of the time points

subplot(3,2,k),mesh(gates\_model(:,:,k));

title(sprintf('Time Point %g',time\_points(k)));

ylabel('Subject');

xlabel('Feature');

end

suptitle('Modeled Data');

% error in model: residual

res = tensoring - gates\_model;

figure % plot residuals

for k = 1:time\_pt % for each of the time points

subplot(3,2,k),mesh(res(:,:,k));

title(sprintf('Time Point %g',time\_points(k)));

ylabel('Subject');

xlabel('Feature');

end

suptitle('Residuals');

Create a plot of the loadings for each component in each mode to see which features, subjects, and time, points are more important in their components and respective modes

Taken From: TensorExplorationPARAFAC.m, Section 11

Input: modeled PARAFAC tensor

What it does: (see title of section)

The MATLAB code is:

%% Components/Loadings

figure

plotfac(factors);

suptitle('Component Matrices for Each Mode');

Create a plot of the scores for each mode to see where each point falls

Taken From: TensorExplorationPARAFAC.m, Section 12

Input: modeled PARAFAC tensor

What it does: (see title of section)

The MATLAB code is:

%% Loadings/Scores

% subjects

figure

subplot(2,2,1),plot(A(:,1),A(:,2),'.');

title('Mode 1')

xlabel('Score 1')

ylabel('Score 2')

%The text command print

% mode1\_labels = num(ia,2);

% for i = 1:size(A,1)

% cc = text(A(i,1),A(i,2),num2str(mode1\_labels(i)));

% end

% features

% mode2\_labels = {'weight','height','bmi','waz','haz','baz','whz','sexn',...

% 'feedingn','gagebirth','birthwt','birthlen','apgar1','apgar5','mage','mracen',...

% 'mmaritn','mcignum','parity','gravida','meducyrs','sesn'}; % former

mode2\_labels = {'weight','height','bmi','waz','haz','baz','whz','sexn',...

'feedingn','gagebirth','apgar1','apgar5','mage','mracen',...

'mcignum','parity','gravida','meducyrs','sesn'}; % yener

subplot(2,2,2),plot(B(:,1),B(:,2),'.');

title('Mode 2')

xlabel('Score 1')

ylabel('Score 2')

for i = 1:size(B,1)

cc = text(B(i,1),B(i,2),mode2\_labels{i});

end

% time

mode3\_labels = {'1','123','366','1462','2558'};

subplot(2,2,3),plot(C(:,1),C(:,2),'.');

title('Mode 3')

xlabel('Score 1')

ylabel('Score 2')

for i = 1:size(C,1)

cc = text(C(i,1),C(i,2),mode3\_labels{i});

end

Create a gradient for each of the characteristics by their number values on the score plot for subjects

Taken From: TensorExplorationPARAFAC.m, Section 13

Input: modeled PARAFAC tensor

What it does: (see title of section)

The MATLAB code is:

% yener tensor

%{

mode2\_labels = {'sexn',...

'feedingn','gagebirth','apgar1','apgar5','mage','mracen',...

'mcignum','parity','gravida','meducyrs','sesn','geniq','sysbp','diabp'};

iqs = chara(:,20);

for k = 9:size(gates\_keep,2)

figure

scatter(A(:,1),A(:,2), 5, gates\_keep(:,k,1));

set(gca,'CLim',[min(gates\_keep(:,k,1)) max(gates\_keep(:,k,1))]);

title(sprintf('%g: %s',k,mode2\_labels{k-8}))

colorbar;

ylabel('Second Component')

xlabel('First Component')

end

%}

Create a plot of the scores colored by SGA to see where they lie

Taken From: TensorExplorationPARAFAC.m, Section 14

Input: modeled PARAFAC tensor

What it does: (see title of section)

The MATLAB code is:

%% Looking at where SGA kids fall

% on the scatter plot model

figure

% colors = colormap(jet(max(num(:,32))-min(num(:,32)+1)));

gscatter(A(:,1),A(:,2), SGA(:,2));

% set(gca,'CLim',[min(iqs) max(iqs)]);

% colorbar

title('Components Colored by SGA')

xlabel('Component 1')

ylabel('Component 2')

Create a heat map for each mode to see where the scores in each plot relative to each other

Taken From: TensorExplorationPARAFAC.m, Section 15

Input: modeled PARAFAC tensor

What it does: (see title of section)

The MATLAB code is:

%% Imagesc

% heat maps of each mode

figure

subplot(2,2,1),imagesc(A),colorbar

title('Mode 1')

ylabel('Subjects')

xlabel('Components')

subplot(2,2,2),imagesc(B),colorbar

title('Mode 2')

ylabel('Features')

xlabel('Components')

subplot(2,2,3),imagesc(C),colorbar

title('Mode 3')

ylabel('Time Points')

xlabel('Components')

Create a plot that shows the characteristics versus the more important component, depending on the results of the PARAFAC model

Taken From: TensorExplorationPARAFAC.m, Section 23

Input: modeled PARAFAC tensor

What it does: (see title of section)

The MATLAB code is:

%% Characteristics vs First Component

[sorting,order] = sortrows(A,2);

labels = {'siteid','sexn',...

'feedingn','gagebirth','birthwt','birthlen','apgar1','apgar5','mage','mracen',...

'mmaritn','mcignum','parity','gravida','meducyrs','sesn','geniq','sysbp','diabp'};

%{

% normal

figure

for k = 1:12

subplot(3,4,k),plot(sorting(:,1),chara(order,k+1),'.');

title(sprintf('%s vs First Component',labels{k}))

ylabel(labels{k})

xlabel('First Component')

end

figure

for k = 1:7

subplot(3,3,k),plot(sorting(:,1),chara(order,k+13),'.');

title(sprintf('%s vs First Component',labels{k+12}))

ylabel(labels{k+12})

xlabel('First Component')

end

%}

%yener

labels = {'sexn',...

'feedingn','gagebirth','apgar1','apgar5','mage','mracen',...

'mcignum','parity','gravida','meducyrs','sesn','geniq','sysbp','diabp','SGA'};

figure

for k = 1:12

subplot(3,4,k),plot(sorting(:,2),chara(order,k+8),'.');

title(sprintf('%s vs Second Component',labels{k}))

ylabel(labels{k})

xlabel('First Component')

end

figure

for k = 1:4

subplot(3,3,k),plot(sorting(:,2),chara(order,k+20),'.');

title(sprintf('%s vs Second Component',labels{k+12}))

ylabel(labels{k+12})

xlabel('First Component')

end

%}

Fuzzy C-Means Clustering

Taken From: TensorExplorationPARAFAC.m, Section 16

Input: first mode of modeled PARAFAC tensor

What it does: clusters the subjects by the first mode of the modeled PARAFAC tensor using Fuzzy C-Means clustering

The MATLAB code is:

%% Fuzzy C-Means Clustering

nC = 5;

analyze = gates\_mtx(:,47);

[VC,UF,~,G,idx] = FuzzyCMeans(analyze,(1:38508)',nC);

K-Means Clustering

Taken From: TensorExplorationPARAFAC.m, Section 16

Input: first mode of modeled PARAFAC tensor

What it does: clusters the subjects by the first mode of the modeled PARAFAC tensor using K-Means clustering

The MATLAB code is:

nC = 5;

analyze = gates\_mtx(:,47);

% idx = kmeans(analyze,nC,'replicates',10)

Choose method with more **explanatory** clusters

Taken From: N/A

Input: results of clustering analysis

What it does: this is evaluated after all clustering analysis within the context of the problem

The MATLAB code is: N/A

Silhouette Plot

Taken From: TensorExplorationPARAFAC.m, Section 17

Input: first mode of modeled PARAFAC tensor, cluster number for subjects

What it does: creates a silhouette plot based on the first mode of the modeled PARAFAC tensor and the cluster number for each subject

The MATLAB code is:

%% Silhouette Plot

% visualizing fuzzy c-means

% silhouette graph

figure

silhouette(analyze,idx);

% set(get(gca,'Children'),'FaceColor',[.8 .8 1])

title ('Silhouette Values per Cluster (Fuzzy C-Means)')

xlabel('Silhouette Value')

ylabel('Cluster')

Silhouette Plot Decision Sequence

Taken From: TensorExplorationPARAFAC.m, Section 17

Input: silhouette plots

What it does: (decision sequence)

i. If the silhouette values are high and positive for each cluster, then the clusters are valid and each cluster is distinct.

ii. If the silhouette values are low and negative for any cluster, then the clusters are not distinct and subjects from one cluster should be in another.

The MATLAB code is: N/A

Create a plot of the PARAFAC scores colored by each cluster to see where each cluster lies

Taken From: TensorExplorationPARAFAC.m, Section 18

Input: first mode of modeled PARAFAC tensor, cluster number for subjects

What it does: (see title of section)

The MATLAB code is:

%% PCA (colored clusters)

% cluster colored PCA plot

figure

hold on

grid off

gscatter(analyze(:,1),analyze(:,1),idx)

title('Scores Scatter Plot with Colored Clusters (Fuzzy C-means)');

xlabel('Component 1');

ylabel('Component 2');

hold off

Find the characteristics of each cluster by finding the means for each characteristic and testing whether these results are significant at 95% level

Taken From: TensorExplorationPARAFAC.m, Sections 19 and 20

Input: characteristics of all the subjects, cluster number for subjects

What it does: (see title of section)

The MATLAB code is:

%% Determining Characteristics of Clusters

% finding the averages for each characteristic

avg\_char = zeros(nC,size(chara,2)); % preallocating

for k = 1:nC

avg\_char(k,:) = nanmean(chara(idx == k,:)); % average values

end

%% Significance testing

combo = combnk(1:nC,2); % the different combinations of cluster testing

tests = zeros(size(combo,1),size(chara,2)+1); % is 95% test sucessful?

pval = zeros(size(combo,1),size(chara,2)+1); % pvalue corresponding to ttest

testcount = 1;

for k = combo' % iterating through different combinations of clusters

for p = 1:size(chara,2); % iterating through different criteria

group1 = chara(idx==k(1),p); % first cluster

group1(isnan(group1)) = []; % take away any nan values

group2 = chara(idx==k(2),p); % second cluster

group2(isnan(group2)) = []; % take away any nan values

% difference in means test

[tests(testcount,p+1),pval(testcount,p+1)] = ttest2(group1,group2);

end

testcount = testcount + 1;

end

tests(:,1:2) = combo;

pval(:,1:2) = combo;

Create a histogram to look at the distributions of characteristics, especially IQ, for each cluster

Taken From: TensorExplorationPARAFAC.m, Section 22

Input: characteristics of all the subjects, cluster number for subjects

What it does: (see title of section)

The MATLAB code is:

% yener

figure

j = 1;

for k = [5 4 2 1 3]

subplot(nC,1,j)

hist(chara(idx == k,20),7)

xlim([0 100])

title(sprintf('Cluster %g',k))

xlabel(sprintf('Gestational Age at Birth'))

ylabel(sprintf('Number of Subjects'))

% axis tight

j=j+1;

end

% suptitle('IQ Distributions')