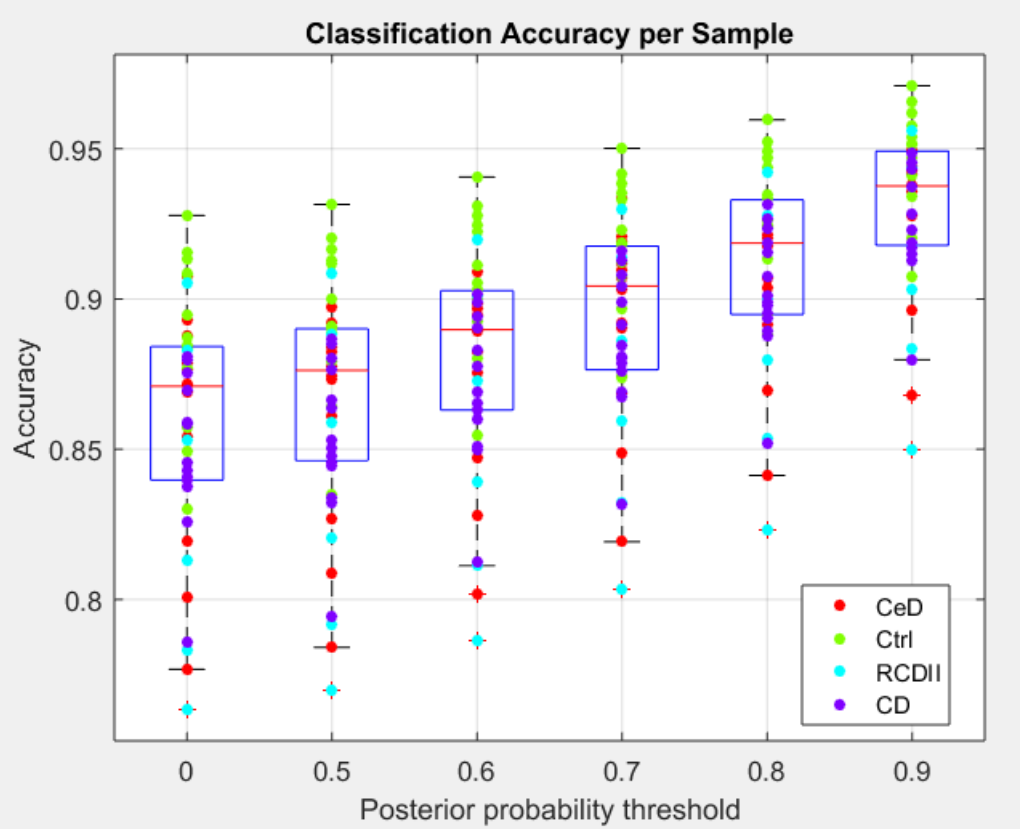
HMIS-2 Dataset Classification

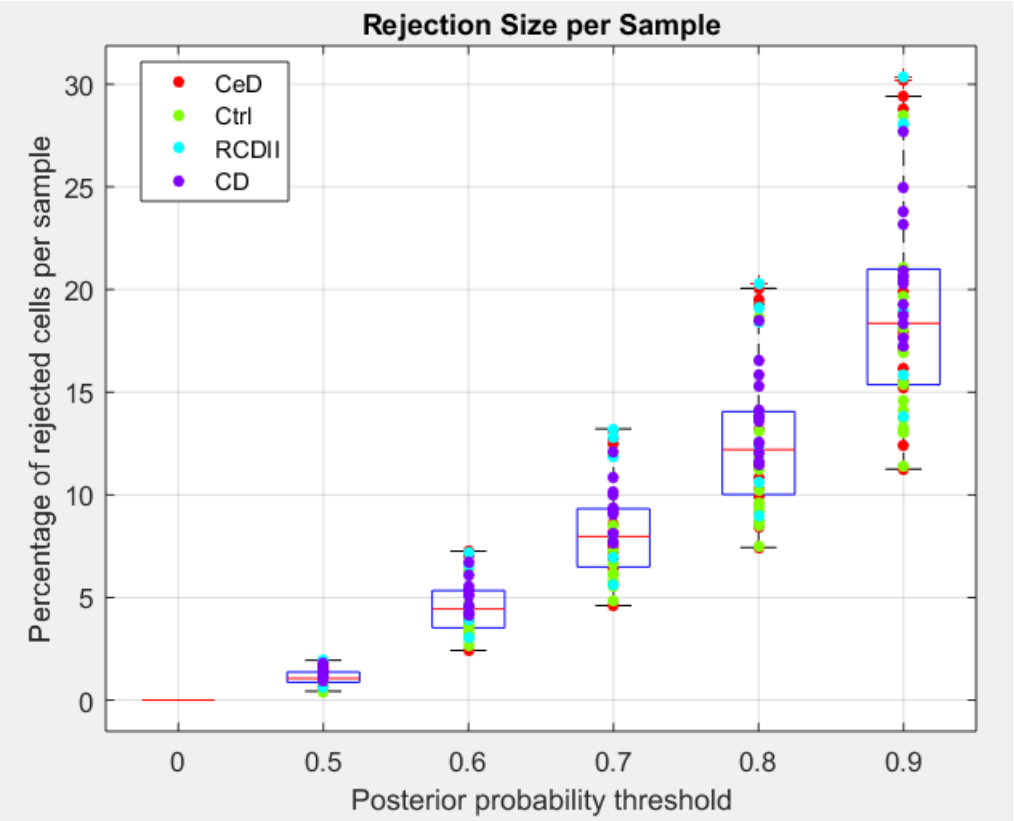
```
%% Read the Data and Preprocess

>> VarNames = {'CCR6','CD19','CKIT','CD11b','CD4','CD8a', ...
   'CD7','CD25','CD123','TCRgd','CD45','CRTH2','CD122', ...
   'CCR7','CD14','CD11c','CD161','CD127','CD8b','CD27', ...
   'IL-15Ra','CD45RA','CD3','CD28','CD38','NKp46','PD-1','CD56'};

>> Samples_Tag = [cellstr(repmat('CeD',4,1)); ...
   cellstr(repmat('Ctrl',7,1)); cellstr(repmat('CeD',9,1));...
   cellstr(repmat('Ctrl',7,1)); cellstr(repmat('RCDII',6,1));...
   cellstr(repmat('CD',14,1))];

>> SamplesData=struct('Data',[],'Labels',{});
>> H=dir(fullfile('Samples\', '*.csv'));
>> SamplesFiles = cellstr(char(H(1:end).name));

>> H=dir(fullfile('Labels\', '*.csv'));
>> LabelsFiles = cellstr(char(H(1:end).name));
>> clear H

>> for i=1:length(SamplesFiles)
>>     SamplesData(i).Data = csvread(['Samples\' SamplesFiles{i}]);
>>     SamplesData(i).Labels = table2cell(readtable(['Labels\'...
      LabelsFiles{i}],'ReadVariableNames',0,'Delimiter',','));
>> end
>> clear i SamplesFiles LabelsFiles

>> Labels = [];
>> for i=1:length(SamplesData)
>>     Labels = [Labels; SamplesData(i).Labels];
>> end
>> CellTypes = unique(Labels);

% remove cells annonated as 'Discard'
%(very small cell types < 0.1% of the total number of cells)
>> CellTypes(strcmp('Discard',CellTypes)) = [];
>> clear i Labels

% Data is already arcsinh(5) transformed

%% run LDA Classifier with 3-fold cross-validation on samples

>> CVO = cvpartition(1:1:length(SamplesData),'k',3);
>> Rejection_Threshold = 0.5:0.1:0.9;
>> Accuracy = zeros(length(SamplesData),1);
>> Accuracy_Rejection = zeros(length(SamplesData), ...
   length(Rejection_Threshold));
>> Rejection_size = zeros(length(SamplesData), ...
   length(Rejection_Threshold));
>> training_time = zeros(CVO.NumTestSets,1);
>> testing_time = zeros(length(SamplesData),1);
>> ConfusionMat = zeros(length(CellTypes));
```

```
>> for i = 1:CVO.NumTestSets
>>     trIdx = find(CVO.training(i));
>>     teIdx = find(CVO.test(i));

>>     DataTrain=[];
>>     LabelsTrain=[];
>>     for j=1:length(trIdx)
>>         DataTrain = [DataTrain; SamplesData(trIdx(j)).Data];
>>         LabelsTrain = [LabelsTrain;SamplesData(trIdx(j)).Labels];
>>     end
>>     clear j
>>     DataTrain(strcmp('Discard',LabelsTrain),:) = [];
>>     LabelsTrain(strcmp('Discard',LabelsTrain)) = [];

>>     tic
>>     classificationLDA = fitcdiscr(...
>>         DataTrain, ...
>>         LabelsTrain);
>>     training_time(i)=toc;          %in seconds

>>     for j=1:length(teIdx)
>>         DataTest = SamplesData(teIdx(j)).Data;
>>         LabelsTest = SamplesData(teIdx(j)).Labels;
>>         tic
>>         [Predictor,scores] = predict(classificationLDA,DataTest);
>>         testing_time(teIdx(j))=toc;           %in seconds
>>         Current_Scores = max(scores,[],2);
>>         Predictor(strcmp('Discard',LabelsTest)) = [];
>>         Current_Scores(strcmp('Discard',LabelsTest)) = [];
>>         LabelsTest(strcmp('Discard',LabelsTest)) = [];
>>         Accuracy(teIdx(j)) = nnz(strcmp(Predictor,LabelsTest))...
          /size(LabelsTest,1);
>>         ConfusionMat = ConfusionMat + confusionmat(LabelsTest,...
          Predictor,'order',CellTypes);

>>         for r=1:length(Rejection_Threshold)
>>             Rejection_size(teIdx(j),r)=nnz(Current_Scores...
              < Rejection_Threshold(r))/size(LabelsTest,1);
>>             Accuracy_Rejection(teIdx(j),r) = nnz(...
              strcmp(Predictor(Current_Scores >= ...
              Rejection_Threshold(r)),LabelsTest(...
              Current_Scores >= Rejection_Threshold(r))))...
              /size(LabelsTest(Current_Scores >= ...
              Rejection_Threshold(r)),1);

>>         end
>>         clear r
>>     end
>>     clear j
>> end

>> Total_time = sum(training_time)+sum(testing_time);
>> training_time = mean(training_time);
>> testing_time = mean(testing_time);
>> cvAcc = mean(Accuracy)*100;
```

```
>> cvSTD = std(Accuracy)*100;
>> disp(['LDA Accuracy = ' num2str(cvAcc) ' ' char(177) ' ' ...
   num2str(cvSTD) ' %'])

LDA Accuracy = 86.1118 ± 3.8557 %

>> clear i Predictor classificationLDA trIdx teIdx CVO DataTrain
   LabelsTrain
>> clear DataTest LabelsTest

%% Accuracy and Rejection size with different rejection thresholds
% Fig. 2
>> figure,
>> boxplot([Accuracy Accuracy_Rejection],'Labels', ...
   {'0','0.5','0.6','0.7','0.8','0.9'}),hold on
>> gscatter([ones(length(Accuracy),1); ...
   2*ones(length(Accuracy_Rejection),1);...
   3*ones(length(Accuracy_Rejection),1); ...
   4*ones(length(Accuracy_Rejection),1);...
   5*ones(length(Accuracy_Rejection),1); ...
   6*ones(length(Accuracy_Rejection),1)],...
   [Accuracy; Accuracy_Rejection(:)],repmat(Samples_Tag,6,1))
>> title('Classification Accuracy per Sample')
>> xlabel('Posterior probability threshold'),ylabel('Accuracy')
>> box on, grid on
```

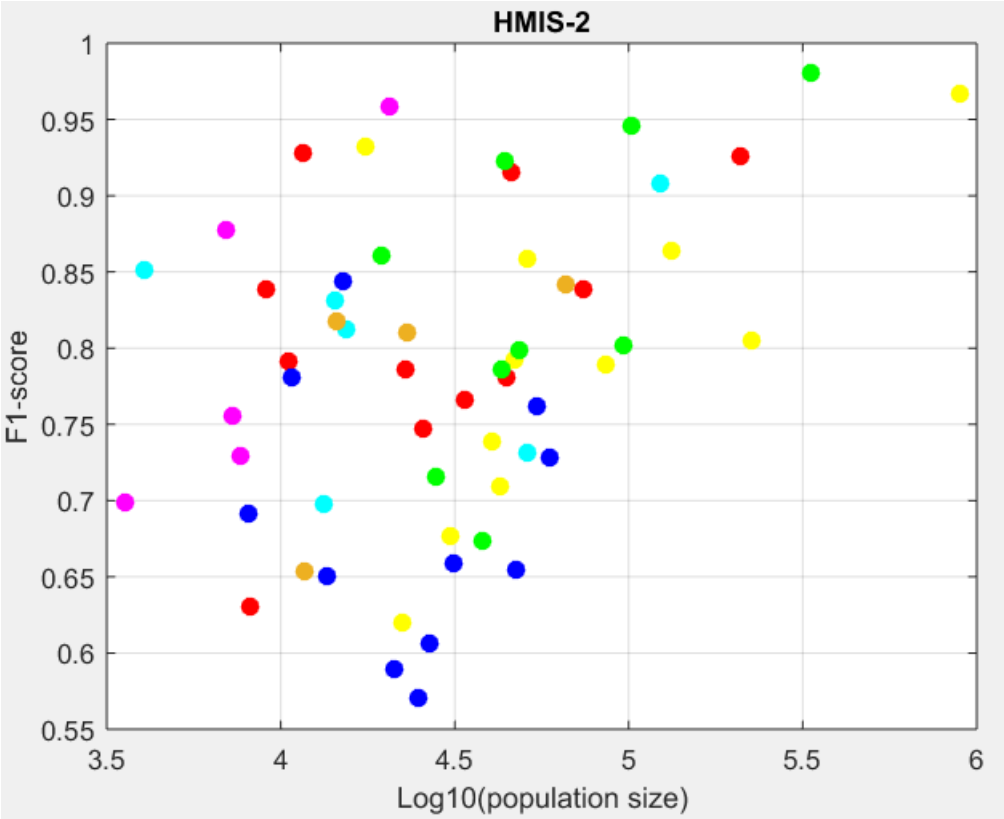


```
>> figure,
>> boxplot([zeros(length(Accuracy),1) Rejection_size*100], ...
   'Labels',{'0','0.5','0.6','0.7','0.8','0.9'}),hold on
>> gscatter([2*ones(length(Rejection_size),1);...
   3*ones(length(Rejection_size),1); ...
   4*ones(length(Rejection_size),1);...
   5*ones(length(Rejection_size),1); ...
   6*ones(length(Rejection_size),1)],...
   Rejection_size(:)*100,repmat(Samples_Tag,5,1))
>> title('Rejection Size per Sample')
>> xlabel('Posterior probability threshold')
>> ylabel('Percentage of rejected cells per sample')
>> box on, grid on
```



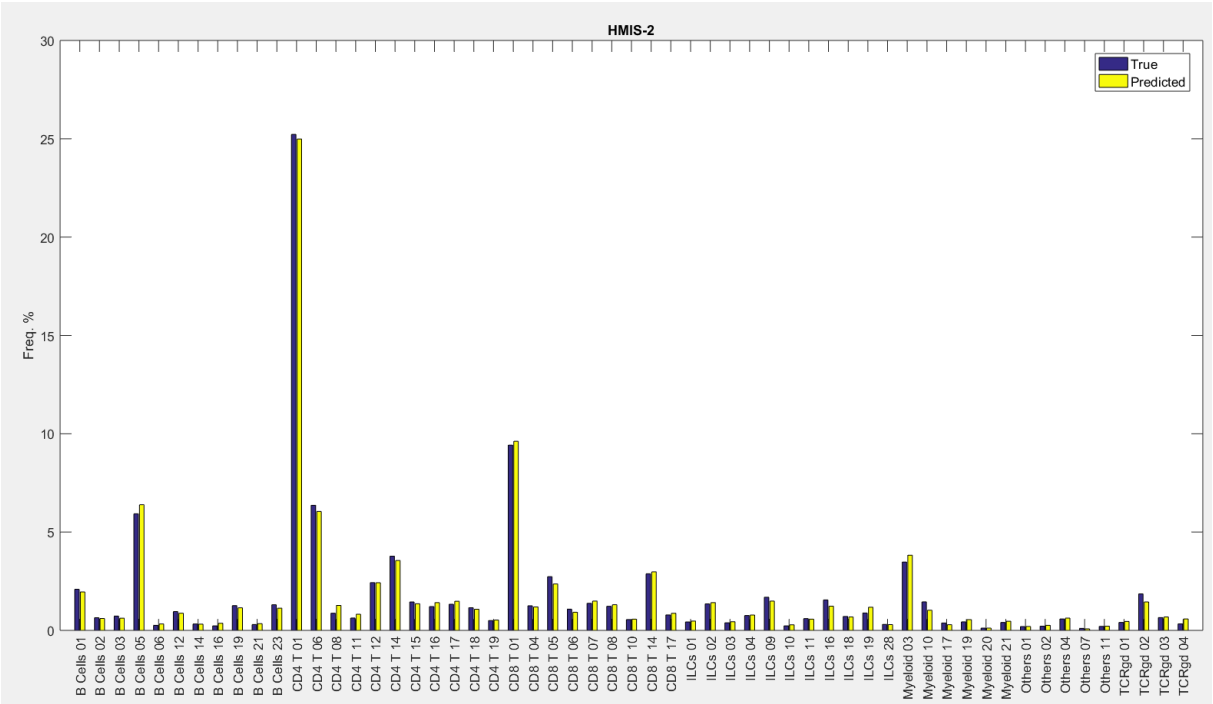

```
%% Performance evaluation
% F1 measure
>> Precision = diag(ConfusionMat)./sum(ConfusionMat,1)';
>> Recall = diag(ConfusionMat)./sum(ConfusionMat,2);
>> Fmeasure = 2 * (Precision.*Recall)./(Precision+Recall);
>> MedianFmeasure = median(Fmeasure);
>> Subset_size = sum(ConfusionMat,2);
>> WeightedFmeasure = (Subset_size./sum(Subset_size))'*Fmeasure;

>> disp(['Median F1-score = ' num2str(MedianFmeasure)])

Median F1-score = 0.78939
```

```
% Supplementary Figure 8A
>> Cmap = [repmat([1 0 0],11,1); repmat([1 1 0],11,1); ...
   repmat([0 1 0],9,1); repmat([0 0 1],11,1); ...
   repmat([0 1 1],6,1); repmat([1 0 1],5,1); ...
   repmat([0.93 0.69 0.13],4,1)];
>> figure,
>> scatter(log10(Subset_size),Fmeasure,50,Cmap,'filled')
>> title('HMIS-2')
>> xlabel('Log10(population size)'),ylabel('F1-score')
>> box on, grid on
```

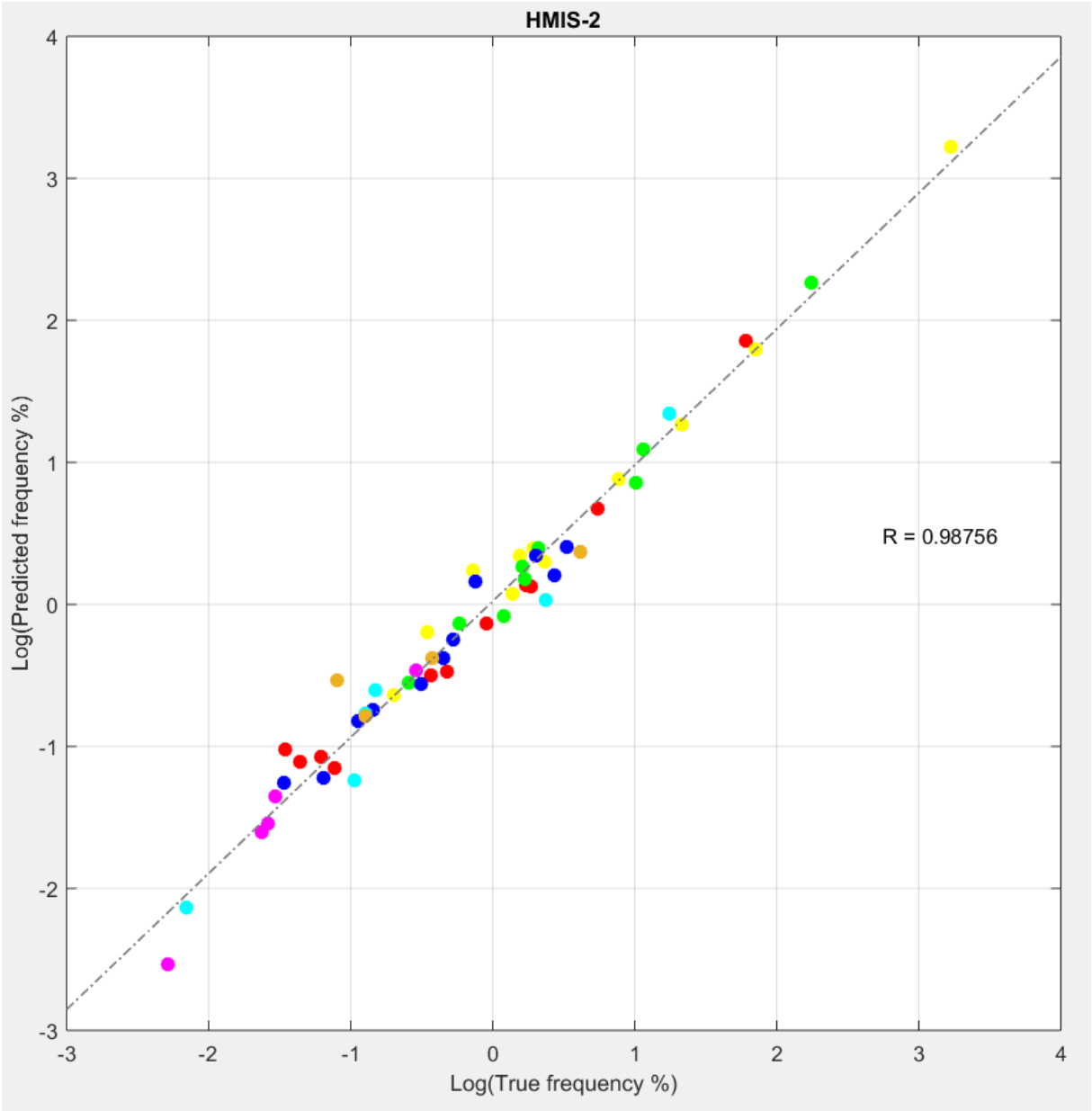



```
%% Population Frequency

>> True_Freq = sum(ConfusionMat,2)./sum(sum(ConfusionMat));
>> Predicted_Freq = sum(ConfusionMat,1)'./sum(sum(ConfusionMat));
>> Max_Freq_diff = max(abs(True_Freq-Predicted_Freq))*100;

>> disp(['delta_f = ' num2str(Max_Freq_diff)])

delta_f = 0.4616

>> figure,bar([True_Freq*100 Predicted_Freq*100])
>> xticks(1:57)
>> xticklabels(CellTypes)
>> xtickangle(90)
>> set(gca,'FontSize',10)
>> set(gca,'XLim',[0 58])
>> legend({'True','Predicted'},'FontSize',10)
>> legend show
```

```
>> ylabel('Freq. %'),title('HMIS-2')
```




```
%% Population Frequency scatter plot

>> Cmap = [repmat([1 0 0],11,1); repmat([1 1 0],11,1); repmat([0 1
>> 0],9,1);...
>> repmat([0 0 1],11,1); repmat([0 1 1],6,1); repmat([1 0 1],5,1);
>> repmat([0.93 0.69 0.13],4,1)];
>> X=log(True_Freq*100);
>> Y=log(Predicted_Freq*100);
>> figure,scatter(X,Y,50,Cmap,'filled')
>> box on, grid on
>> xlabel('Log(True frequency %)')
>> ylabel('Log(Predicted frequency %)')
>> title('HMIS-2')
>> % for k=1:length(CellTypes)
>> % text(X(k),Y(k),CellTypes{k})
>> % end
>> lsline
>> text(0,0,['R = ' num2str(corr(X,Y))])
```


HMIS-2
Log(Predicted frequency %)
Log(True frequency %)
R = 0.98756
4
3
2
1
0
-1
-2
-3
-3
-2
-1
0
1
2
3
4