checkModel.m

Summary: Compute goodness-of-fit measures to assess the predictive power of the model. If no test dataset is provided, measure on train dataset.

Input arguments:

```
model           model to be assessed
test_x          (optional) test inputs to try model on
test_y          (optional) test outputs to try model on
```

Code

```
function checkModel( model, test_x, test_y )

    if nargin == 1; x = model.inputs; y = model.targets;
    else x = test_x; y = test_y; end;

    % 1. Gather information about the model and the test dataset
    [m, n] = size(x);
    [m1, d] = size(y);
    if m1~=m; return; end;
    LP = zeros(m, 1);
    sampleSize = 1000;
    sampleLP = zeros(m, 1);
    predictions = zeros(m, d);

    % 2. Compute model and optimal NLPD
    for j=1:m
        [mu, sigma] = model.fcn(model, x(j,:)', zeros(n));
        %disp(mu);
        LP(j) = NLPD(y(j,:)', mu, sigma, d);
        sample = mvnrnd(mu, sigma, sampleSize);
        aux = cellfun(@(v) NLPD(v', mu, sigma, d), num2cell(sample, 2));
        sampleLP(j) = mean(aux);
        predictions(j, :) = mu;
    end
    fprintf('NLPD of model\n \t True \t Optimal \n Mean %f  %f \n Std  %f  %f\n',m

    % 3. Compute relative error
    diff = abs((predictions-y)./y);
    relDiff = mean(100*diff)';
    fprintf('Mean relative difference of each variable (%%):\n');
    disp(relDiff);
    fprintf('Total average relative difference is %f%%\n', mean(relDiff));


    % 4. Check SNR's
    hyp = model.hyp;
    SNR = exp(hyp(end-1,:)-hyp(end,:));
    if any(SNR > 500)
        fprintf('SNR is greater than 500 for variables %s\n', num2str(find(SNR>500
```

```
    else
        fprintf('All SNRs are less than 500.\n');
    end

    % 5. Check log-signal-std
    lsstd = hyp(end-1, :);
    if any(lsstd<log(0.1))
        low = lsstd<log(0.1);
        fprintf('LSSTD is lower than log(0.1) for\n variables \t%s\n values \t%s\n
    end
    if any(lsstd>log(10))
        high = lsstd>log(10);
        fprintf('LSSTD is higher than log(10) for\n variables \t%s\n values \t%s\n
    end
    if all(and(lsstd>log(0.1), lsstd<log(10)))
        fprintf('All log-signal-std are within safe range.\n');
    end

    % 6. Check lengthscales
    lengthScale = exp(hyp(1:end-2,:));
    inputStd = repmat(std(model.inputs)',[1,d]);
    fprintf('LengthScale/std ratios:\n');
    disp(lengthScale./inputStd);


end
```

Published with MATLAB® R2014a