computeLogLikelihood

Comuputes the marginal log likelihood as evidence of the current trained model parameter by solving the equation log p(y|X, alpha, log|Ky|) = -1/2 * (y - m)T * alpha - 1/2 log|Ky| - N/2 log(2pi) where alpha is the inverse matrix product of alpha = Ky^-1 * (y - m).

Syntax

```
computeLogLikelihood(y, m, alpha, logDet, N)
```

Description

computeLogLikelihood(y, m, alpha, logDet, N)

Input Argurments

y column vector of regression targets.

m column vector of regression means.

y column vector of regression weights.

logDet real scalar of K matrix log determinate.

N number of observations. Number of reference angles.

Output Argurments

lml real scalar of log marginal likelihood.

Requirements

- Other m-files required: None
- Subfunctions: None
- MAT-files required: None

See Also

- decomposeChol
- computeAlphaWeights
- initKernelParameters


```
function lml = computeLogLikelihood(y, m, alpha, logDet, N)
    arguments
        % validate inputs as real column vectors
        y (:,1) double {mustBeReal, mustBeVector}
        % m can be zero if zero gpr runs
        m (:,1) double {mustBeReal, mustBeVector}
        alpha (:,1) double {mustBeReal, mustBeVector, mustBeEqualSize(y, alpha)}
        % validate inputs as real scalar
        logDet (1,1) double {mustBeReal}
        N (1,1) double {mustBeReal}
    end

    % get residual of targets and mean
    residual = y - m;
```

```
    % compute log marginal likelihood
    lml = -0.5 * (residual' * alpha + logDet + N * log(2 * pi));
end

% Custom validation functions
function mustBeEqualSize(a, b)
    if ~isequal(length(a), length(b))
        eid = 'Size:notEqual';
        msg = 'Vectors must be the same length.';
        throwAsCaller(MException(eid,msg))
    end
end
```

Published with MATLAB® R2020b