User Guide for the SepGP Package

Contents

I. Description

The SepGP package is designed to model a spatio-temporal function represented by

$$f : (x, t) \in \mathbb{R}^d \times \mathbb{R} \mapsto f(x, t) \in \mathbb{R},$$

and assumed to be observed for each point $x \in \mathbb{R}$ at discretization times $t_1, \dots, t_{N_t}$.

The main features of the SepGP package include:

- **Model Construction**: Create Gaussian Process models with separable covariance functions tailored to the data structure.
- **Parameter Optimization**: Use a Leave-One-Out cross-validation method combined with maximum likelihood estimation to optimize model parameters.
- **Prediction**: Make predictions for new spatial points and all discretization times $t_1, \dots, t_{N_t}$.

II. Main Functions

a) SepGP: Separable Gaussian Process Object

The SepGP function allows the creation of an object of class SepGP, initializing a Gaussian process model with customizable prior covariance and mean structures, along with an optional nugget effect.

Usage `SepGP(D, FD, s, covtype, phi, nugget)`

Parameters

- `D`: Design of experiments containing the spatial points.
- `FD`: Matrix of function evaluations at **D**.
- `s`: Degree of the polynomial for the prior mean function.
- `B`: (Optional) Matrix of regression coefficients for the prior mean. If not specified, a matrix consisting of ones is used by default.
- `covtype`: Type of spatial covariance function, with a default value of `"gauss"`. Acceptable values are `"gauss"`, `"exp"`, `"matern3_2"`, and `"matern5_2"`.
- `phi`: (Optional) Hyperparameter vector for the covariance function. If not specified, a vector of ones is used by default.
- `philow`: (Optional) Lower bounds for the hyperparameter `phi`. If not specified, a vector consisting of `0.1` is used by default.
- `phiup`: (Optional) Upper bounds for the hyperparameter `phi`. If not specified, a vector consisting of `10` is used by default.
- `Sigmat`: (Optional) Temporal covariance matrix (for discretization times). If not specified, an identity matrix of appropriate size ($N_t \times N_t$) is used by default.
- `nugget`: (Optional) A small positive value added to the diagonal of the covariance matrix to improve numerical stability, with a default value of `1e-16`.

Value

- `gp`: An object of class `SepGP` representing a Gaussian process model with the specified parameters.

b) Covmat: Calculate the Covariance Matrix for SepGP

This function computes the covariance matrix between two design of experiments based on a `SepGP` model.

Usage `Covmat(object, D1, D2)`

Parameters

- `object`: An object of class `SepGP`.
- `D1`: First Design of Experiments.
- `D2`: Second Design of Experiments.

Value

- `Cmat`: Covariance matrix between **D1** and **D2**.

c) PriorMean: Calculate the Prior Mean for SepGP

The `PriorMean` function computes the prior mean for a given design of experiments (`Dnew`) based on an existing `SepGP` model. It constructs a polynomial prior mean function using the coefficients and degree specified in the `SepGP` object.

Usage `PriorMean(object, Dnew)`

Parameters

- `object`: An object of class `SepGP`, representing a Gaussian process model with a separable covariance structure.
- `Dnew`: A matrix representing a new design of experiments (DOE) for which the prior mean will be computed.

Values

- `M`: A matrix representing the prior mean values at each point in **Dnew**.
- `H`: A matrix where each row corresponds to the values of the polynomial basis functions evaluated at a point in `Dnew`.

d) GPoptim: SepGP Model parameters Optimization

The `GPoptim` function optimizes the hyperparameters of a given `SepGP` model. This includes tuning the spatial covariance function parameters, the temporal covariance matrix, and the coefficients matrix associated with the prior mean function. Optimization is achieved using a combination of leave-one-out cross-validation and maximum likelihood estimation methods, providing a robust fit for the model.

Usage `GPoptim(object, maxit)`

Parameters

- `object`: An object of class `SepGP`, representing a Gaussian process model with a separable covariance structure.
- `maxit`: (Optional) An integer specifying the maximum number of iterations for the optimization algorithm (refer to the `optim` function in the `stats` package for details). The default value is 100. You should choose a reasonable value for maxit based on the dimensionality of the hyperparameters of the spatial covariance function.

Values An optimized `SepGP` object with updated hyperparameters:

- `phi`: Optimal values for the covariance hyperparameters.
- `B`: Estimated coefficient matrix derived from the optimized hyperparameters.
- `Sigmat`: Estimated temporal covariance matrix.

e) predictSepGP: Make Predictions Using a SepGP Model

This function computes the posterior mean and covariance matrix for for new spatial points based on a `SepGP` model.

Usage `predictSepGP(object, Dpred, covcompute)`

Parameters

- `object`: An object of class `SepGP`, representing a Gaussian process model with a separable covariance structure.
- `Dpred` : A new design of spatial points for which predictions are to be made.
- `covcompute` : A logical value indicating whether the posterior covariance should be computed. The default value is `TRUE`.

Values A list with the following elements:

- `Mean`: Posterior mean vector of the predictions for each new input in `Dpred`.
- `Covsp`: Spatial posterior covariance matrix of the predictions.
- `Covmat`: Posterior covariance matrix of the predictions, quantifying the uncertainty for each point in `Dpred`.

III. Discussion

The `SepGP` package is particularly effective in cases where separability is justified, such as when a separability test confirms its suitability. However, it currently supports a limited range of covariance functions, specifically isotropic Gaussian, exponential, and Matérn kernels (with smoothness parameters $\nu = 3/2$ and $\nu = 5/2$).

The optimization process in `SepGP` is based on a leave-one-out cross-validation (LOO-CV) approach. While this method is generally reliable, it can be time-consuming, especially for cases with high spatial dimensionality and/or large numbers of discretized time points.

IV. Reference

Perrin, G. (2019). *Adaptive calibration of a computer code with time-series output*. Reliability Engineering and System Safety. Elsevier. Retrieved from https://hal.science/hal-01794807 and PDF link.