2.4 What does *mlegp* do?

The package *mlegp* extends the Gaussian process model of (3) by allowing the user to replace the identity matrix I in equations (3) and (4) with a diagonal matrix N, thereby specifying the *nugget matrix* up to a multiplicative constant. This extension provides some flexibility for modeling heteroscedastic responses. The user also has the option of fitting a GP with a constant mean (i.e., $\mu(\theta) \equiv \mu_0$) or mean functions that are linear regression functions in all elements of θ (plus an intercept term). For multi-dimensional output, the user has the option of fitting independent GPs to each dimension (i.e., each type of observation), or to the most important principle component weights following singular value decomposition. The latter is ideal for data rich situations, such as functional output, and is explained further in Section (5). GP accuracy is analyzed through diagnostic plots of cross-validated predictions and cross-validated residuals, which were described in Section (2.3). Sensitivity analysis tools including FANOVA decomposition, and plotting of main and two-way factor interactions are described in Section (4).

3 Examples: Gaussian process fitting and diagnostics

3.1 A simple example

The function *mlegp* is used to fit Gaussian processes (GPs) to a vector or matrix of responses observed under the same set of design parameters. The example below shows how to fit multiple Gaussian processes to multiple outputs $z1$ and $z2$ for the design matrix x. Diagnostic plots are obtained using the *plot* function, which graphs observed values vs. cross-validated predicted values for each GP. The plot obtained from the code below appears in Figure (1).

```
> x = -5:5
> z1 = 10 - 5 * x + rnorm(length(x))
> z2 = 7 * sin(x) + rnorm(length(x))
> fitMulti = mlegp(x, cbind(z1, z2))
> plot(fitMulti)
```

After the GPs are fit, simply typing the name of the object (e.g., $fitMulti$) will return basic summary information.

```
> fitMulti

num GPs: 2
Total observations (per GP): 11
Dimensions: 1
```

We can also access individual Gaussian processes by specifying the index. The code below, for examples, displays summary information for the first Gaussian process, including diagnostic statistics of cross-validated root mean squared error (CV RMSE) and cross-validated root max squared error (CV RMaxSE), where squared error corresponds to the squared difference between cross-validated predictions and observed values.

```
> fitMulti[[1]]

Total observations = 11
Dimensions = 1

mu = 9.289288
sig2:        240.8926
nugget:        0

Correlation parameters:
```

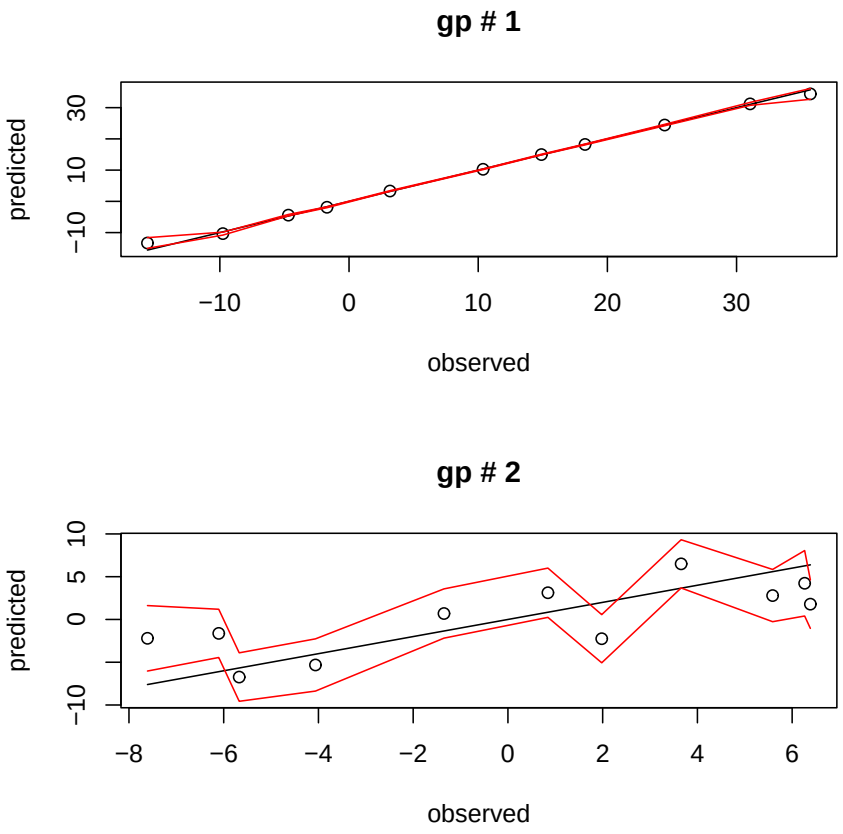

Figure 1: Gaussian process diagnostic plots. Open circles, cross-validated predictions; solid black lines, observed values; solid red lines, confidence bands corresponding to cross-validated predictions ± standard deviation.

```
       beta a
1 0.1317689 2

Log likelihood = -28.34312

CV RMSE: 0.8175645
CV RMaxSE: 5.140437
```

3.2 Heteroscedastic responses and the nugget matrix

In cases where the responses are heteroscedastic (have non-constant variance), it is possible to specify the diagonal nugget matrix up to a multiplicative constant. Future versions of *mlegp* will allow more complicated forms of the nugget matrix; currently, we recommend specifying the nugget matrix based on sample variances for replicate design points (which is easily obtained using the function *varPerReps*), or the use of prior information. In the example below, we demonstrate how to fit a Gaussian process with a constant nugget term and a Gaussian process where the diagonal nugget matrix is specified up to a multiplicative constant. First we generate heteroscedastic data, with variance related to the design parameter.

```
> x = seq(0, 6, by = 0.1)
> z = sin(x) + rnorm(length(x), sd = 0.1 * x)
```

By default, a nugget term is automatically estimated if there are replicates in the design matrix, and is not estimated otherwise. However, one can estimate a nugget term by specifying an initial scalar value for the 'nugget' argument during the call to *mlegp*. This is done in the code below.

```
> fit1 = mlegp(x, z, nugget = (0.1 * mean(x))^2, param.names = "x1")
```

Alternatively, one can set 'nugget' equal to a vector corresponding to the diagonal nugget matrix as described in Section (2.4). This allows the nugget matrix to be specified up to a multiplicative constant, and is demonstrated in the code below.

```
> fit2 = mlegp(x, z, nugget = x^2, param.names = "x1")
```

It is also possible to force a constant nugget term or the diagonal elements of the nugget matrix to have a minimum value by setting the argument 'min.nugget'. This is especially important when the responses are noiseless, and in other situations when the variance-covariance matrix of the GP is not stable otherwise.

Finally, we compare the accuracy of each fitted GP using diagnostic plots. The diagnostic plots are constructed with the argument 'type' equal to '2', which plots parameter values vs. cross-validated predictions. The output from the code is displayed in Figure (2). Note that when a constant nugget term is assumed, confidence bands are overly conservative for low values of $x1$ and are less likely to capture the observed responses for high values of $x1$.

```
> par(mfrow = c(1, 2))
> plot(fit1, type = 2)
> lines(x, sin(x), lty = 2)
> plot(fit2, type = 2)
> lines(x, sin(x), lty = 2)
```

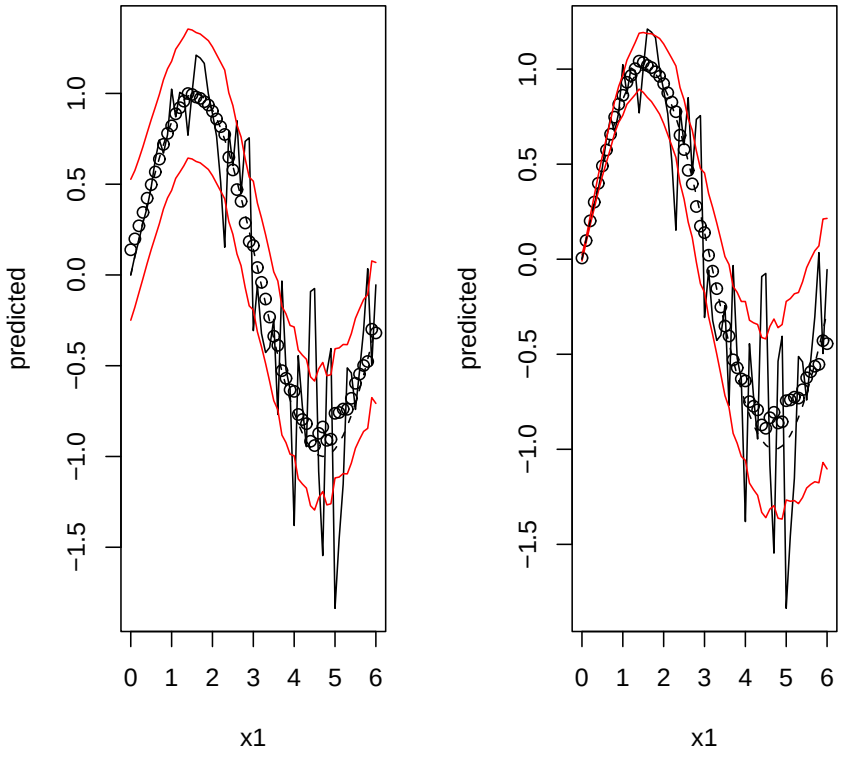

Figure 2: Diagnostic plots for Gaussian processes with constant nugget term (left) and diagonal nugget matrix (right). Open circles, cross-validated predictions; solid black lines, observed response; dotted black lines, the true (noiseless) response; solid red lines, confidence bands.