```
> plot(moto.btlm, main = "Bayesian CART,", layout = "surf")
```

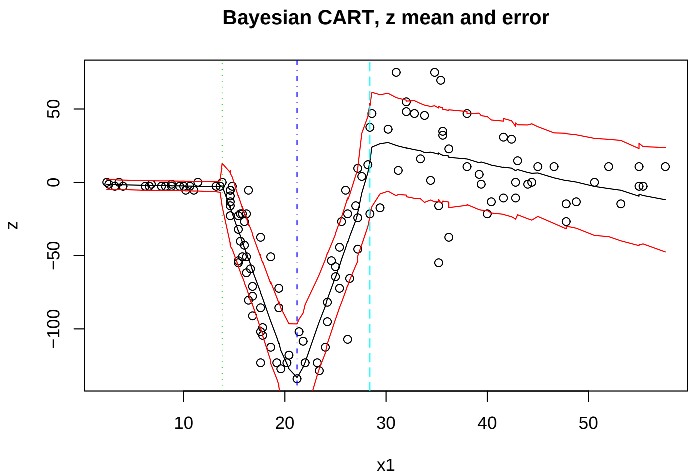


Figure 13: Posterior predictive distribution using `btlm` on the motorcycle accident data: mean and 90% credible interval

left-hand part of the input space is likely linear. One might further hypothesize that the right-hand region is also linear, perhaps with the same mean as the left-hand region, only with higher noise. The `b*` and `tgp` functions can force an i.i.d. hierarchical linear model by setting `bprior=b0`. Moreover, instead of rescaling the responses with `m0r1`, one might try encoding a mixture prior for the nugget in order to explicitly model region-specific noise. This requires direct usage of `tgp`.

```
> p <- tgp.default.params(2)
> p$bprior <- "b0"
> p$nug.p <- c(1, 0.1, 10, 0.1)
> moto.tgp <- tgp(X = mcycle[, 1], Z = mcycle[, 2],
+     params = p, BTE = c(2000, 22000, 2))
```

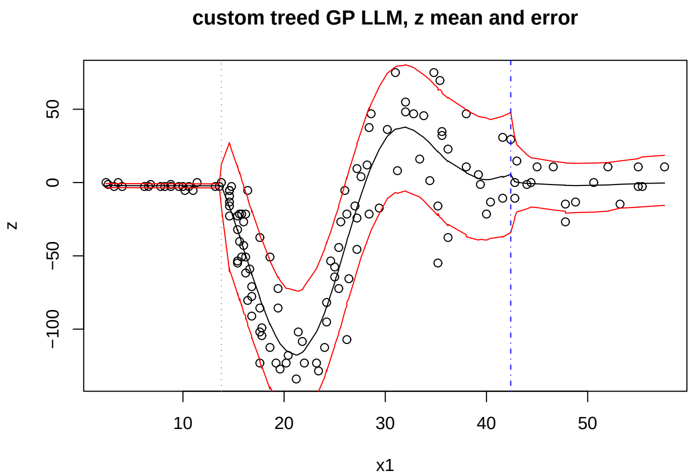
The resulting posterior predictive surface is shown in the *top* half of Figure 14. The *bottom* half of the figure shows the norm (difference) in predictive quantiles, clearly illustrating the treed GP's ability to capture input-specific noise in the posterior predictive distribution.

3.5 Friedman data

This Friedman data set is the first one of a suite that was used to illustrate MARS (Multivariate Adaptive Regression Splines) [9]. There are 10 covariates in the data ($\mathbf{x} = \{x_1, x_2, \dots, x_{10}\}$). The function that describes the responses (Z), observed with standard Normal noise, has mean

$$E(Z|\mathbf{x}) = \mu = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5, \qquad (16)$$

```
> plot(moto.tgp, main = "custom treed GP LLM,", layout = "surf")
```




```
> plot(moto.tgp, main = "custom treed GP LLM,", layout = "as")
```

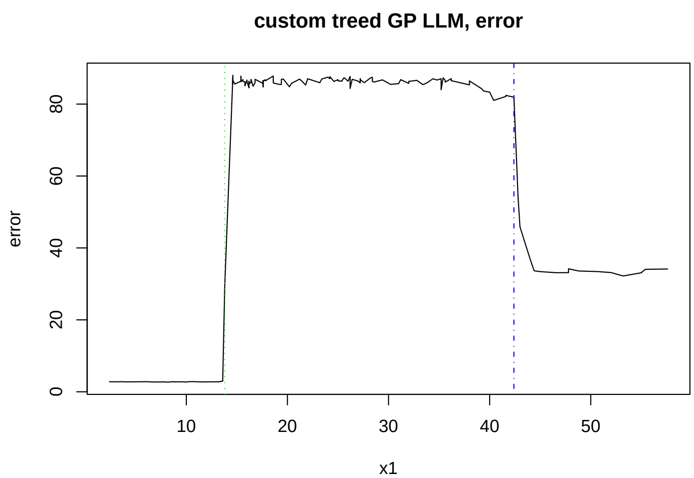


Figure 14: *top* Posterior predictive distribution using a custom parameterized **tgp** call on the motorcycle accident data: mean and 90% credible interval; *bottom* Quantile-norm (90%-5%) showing input-dependent noise.

but depends only on $\{x_1, \ldots, x_5\}$, thus combining nonlinear, linear, and irrelevant effects. Comparisons are made on this data to results provided for several other models in recent literature. Chipman et al. [4] used this data to compare their linear CART algorithm to four other methods of varying parameterization: linear regression, greedy tree, MARS, and neural networks. The statistic they use for comparison is root mean-square error (RMSE)

$$\mathrm{MSE} = \sum_{i=1}^{n}(\mu_i - \hat{z}_i)^2/n \qquad\qquad \mathrm{RMSE} = \sqrt{\mathrm{MSE}}$$

where $\hat{z}_i$ is the model-predicted response for input $\mathbf{x}_i$. The $\mathbf{x}$'s are randomly distributed on the unit interval.

Input data, responses, and predictive locations of size $N = 200$ and $N' = 1000$, respectively, can be obtained by a function included in the `tgp` package.

```
> f <- friedman.1.data(200)
> ff <- friedman.1.data(1000)
> X <- f[, 1:10]
> Z <- f$Y
> XX <- ff[, 1:10]
```

This example compares Bayesian linear CART with Bayesian GP LLM (not treed), following the RMSE experiments of Chipman et al. It helps to scale the responses so that they have a mean of zero and a range of one. First, fit the Bayesian linear CART model, and obtain the RMSE.

```
> fr.btlm <- btlm(X = X, Z = Z, XX = XX, tree = c(0.95,
+     2, 10), m0r1 = TRUE)
> fr.btlm.mse <- sqrt(mean((fr.btlm$ZZ.mean - ff$Ytrue)^2))
> fr.btlm.mse
```

Next, fit the GP LLM, and obtain its RMSE.

```
> fr.bgpllm <- bgpllm(X = X, Z = Z, XX = XX, m0r1 = TRUE)
> fr.bgpllm.mse <- sqrt(mean((fr.bgpllm$ZZ.mean - ff$Ytrue)^2))
> fr.bgpllm.mse
```

So, the GP LLM is 3.868 times better than Bayesian linear CART on this data, in terms of RMSE (in terms of MSE the GP LLM is 1.967 times better). Watching the evolution of the Markov chain for the GP LLM (via the progress statements written to `stdout`, not shown because the not would fit on the page), it is easy to see how the GP LLM quickly learns that $\mathbf{b} = (1, 1, 1, 0, 0, 0, 0, 0, 0, 0)$, and that $\beta_4 \approx 4$ and $\beta_5 \approx 10$—basically that only the first three inputs contribute nonlinearly, the fourth and fifth contribute linearly, and the remaining five not at all [10].

3.6 Adaptive Sampling

In this section, sequential design of experiments, a.k.a. *adaptive sampling*, is demonstrated on the exponential data of Section 3.3. Gathering, again, the data:

```
> exp2d.data <- exp2d.rand()
> X <- exp2d.data$X
> Z <- exp2d.data$Z
> Xcand <- exp2d.data$XX
```