area of research.

3 Examples using tgp

The following subsections take the reader through a series of examples based, mostly, on synthetic data. At least two different `b*` models are fit for each set of data, offering comparisons and contrasts. Duplicating these examples in your own `R` session is highly recommended. The `Stangle` function can help extract executable `R` code from this document. For example, the code for the exponential data of Section 3.3 can be extracted with one command.

```
> Stangle(vignette("exp", package="tgp")$file))
```

This will write a file called "exp.R". Additionally, each of the subsections that follow is available as an `R` demo. Try `demo(package="tgp")` for a listing of available demos. To invoke the demo for the exponential data of Section 3.3 try `demo(exp, package="tgp")`. This is equivalent to `source("exp.R")` because the demos were created using `Stangle` on the source files of this document.

Each subsection (or subsection of the appendix) starts by seeding the random number generator with `set.seed(0)`. This is done to make the results and analyses reproducible within this document, and in demo form. I recommend you try these examples with different seeds and see what happens. Usually the results will be similar, but sometimes (especially when the data (`X, Z`) is generated randomly) they may be quite different.

Other successful uses of the methods in this package include applications to the Boston housing data [14, 13], and designing an experiment for a reusable NASA launch vehicle [11, 12] called the Langely glide-back booster (LGBB).

3.1 1-d Linear data

Consider data sampled from a linear model.

$$z_i = 1 + 2x_i + \epsilon, \quad \text{where} \quad \epsilon_i \overset{\text{iid}}{\sim} N(0, 0.25^2) \tag{15}$$

The following `R` code takes a sample $\{\mathbf{X}, \mathbf{Z}\}$ of size $N = 50$ from (15). It also chooses $N' = 99$ evenly spaced predictive locations $\tilde{\mathbf{X}} = \mathtt{XX}$.

```
> X <- seq(0, 1, length = 50)
> XX <- seq(0, 1, length = 99)
> Z <- 1 + 2 * X + rnorm(length(X), sd = 0.25)
```

Using `tgp` on this data with a Bayesian hierarchical linear model goes as follows:

```
> lin.blm <- blm(X = X, XX = XX, Z = Z)
```

```
burn in:
r=1000 d=[0]; n=50

Sampling @ nn=99 pred locs:
r=1000 d=[0]; mh=1 n=50
r=2000 d=[0]; mh=1 n=50
r=3000 d=[0]; mh=1 n=50
```

```
> plot(lin.blm, main = "Linear Model,", layout = "surf")
> abline(1, 2, lty = 3, col = "blue")
```

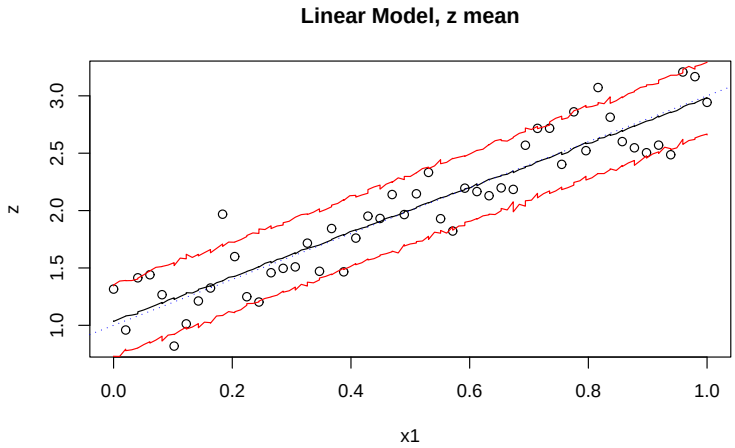

Figure 3: Posterior predictive distribution using `blm` on synthetic linear data: mean and 90% credible interval. The actual generating lines are shown as blue-dotted.

MCMC progress indicators are echoed every 1,000 rounds. The linear model is indicated by `d=[0]`. For `btlm` the MCMC progress indicators are boring, but we will see more interesting ones later. In terminal versions, e.g. `Unix`, the progress indicators can give a sense of when the code will finish. GUI versions of `R`—`Windows` or `MacOS X`—can buffer `stdout`, rendering this feature essentially useless as a real–time indicator of progress. Progress indicators can be turned off by providing the argument `verb=0`. Further explanation on the verbosity of screen output and interpretations is provided in Appendix B.2.

The generic `plot` method can be used to visualize the fitted posterior predictive surface (with option `layout = 'surf'`) in terms of means and credible intervals. Figure 3 shows how to do it, and what you get. The default option `layout = 'both'` shows both a predictive surface and error (or uncertainty) plot, side by side. The error plot can be obtained alone via `layout = 'as'`. Examples of these layouts appear later.

If, say, you were unsure about the dubious "linearness" of this data, you might try a GP LLM (using `bgpllm`) and let a more flexible model speak as to the linearity of the process.

```
> lin.gpllm <- bgpllm(X = X, XX = XX, Z = Z)

burn in:
r=1000 d=[0]; n=50

Sampling @ nn=99 pred locs:
r=1000 d=[0]; mh=1 n=50
r=2000 d=[0]; mh=1 n=50
r=3000 d=[0]; mh=1 n=50


> plot(lin.gpllm, main = "GP LLM,", layout = "surf")
> abline(1, 2, lty = 4, col = "blue")
```

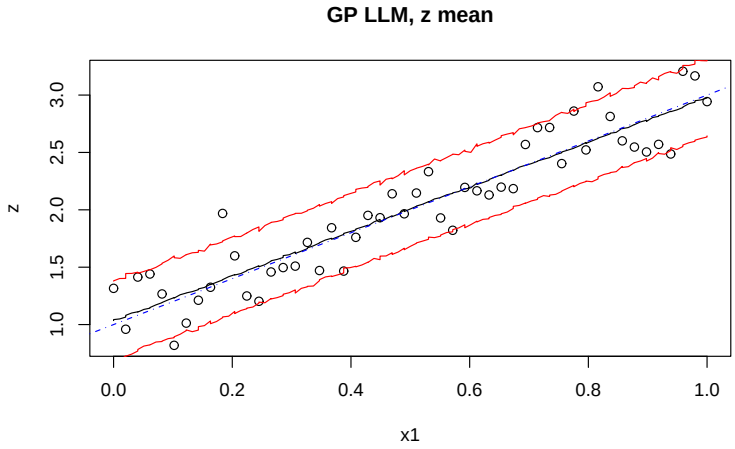

Figure 4: Posterior predictive distribution using `bgpllm` on synthetic linear data: mean and 90% credible interval. The actual generating lines are shown as blue-dotted.

Whenever the progress indicators show `d=[0]` the process is under the LLM in that round, and the GP otherwise. A plot of the resulting surface is shown in Figure 4 for comparison. Since the data is linear, the resulting predictive surfaces should look strikingly similar to one another. On occasion, the GP LLM may find some bendy–ness in the surface. This happens rarely with samples as large as $N = 50$, but is quite a bit more common for $N < 20$.

To see the proportion of time the Markov chain spent in the LLM requires the gathering of traces (Appendix B.1). For example

```
> lin.gpllm.tr <- bgpllm(X = X, XX = 0.5, Z = Z, pred.n = FALSE,
+     trace = TRUE, verb = 0)
> mla <- mean(lin.gpllm.tr$trace$linarea$la)
> mla

[1] 0.96
```

shows that the average area under the LLM is 0.96. Progress indicators are suppressed with `verb=0`. Alternatively, the probability that input location `xx` = 0.5 is under the LLM is given by

```
> 1 - mean(lin.gpllm.tr$trace$XX[[1]]$b1)

[1] 0.96
```

This is the same value as the area under the LLM since the process is stationary (i.e., there is no treed partitioning).

3.2 1-d Synthetic Sine Data

Consider 1-dimensional simulated data which is partly a mixture of sines and cosines, and partly linear.

$$z(x) = \begin{cases} \sin\left(\frac{\pi x}{5}\right) + \frac{1}{5}\cos\left(\frac{4\pi x}{5}\right) & x < 10 \\ x/10 - 1 & \text{otherwise} \end{cases} \tag{16}$$

The `R` code below obtains $N = 100$ evenly spaced samples from this data in the domain $[0, 20]$, with noise added to keep things interesting. Some evenly spaced predictive locations `XX` are also created.

```
> X <- seq(0, 20, length = 100)
> XX <- seq(0, 20, length = 99)
> Z <- (sin(pi * X/5) + 0.2 * cos(4 * pi * X/5)) *
+     (X <= 9.6)
> lin <- X > 9.6
> Z[lin] <- -1 + X[lin]/10
> Z <- Z + rnorm(length(Z), sd = 0.1)
```

By design, the data is clearly nonstationary. Perhaps not knowing this, a good first model choice for this data might be a GP.

```
> sin.bgp <- bgp(X = X, Z = Z, XX = XX, verb = 0)
```

Figure 5 shows the resulting posterior predictive surface under the GP. Notice how the (stationary) GP gets the wiggliness of the sinusoidal region, but fails to capture the smoothness of the linear region. This is because the data comes from a process that is nonstationary.

So one might consider a Bayesian treed linear model (LM) instead.

```
> sin.btlm <- btlm(X = X, Z = Z, XX = XX)

burn in:
**GROW** @depth 0: [0,0.424242], n=(43,57)
**GROW** @depth 1: [0,0.252525], n=(26,19)
**GROW** @depth 2: [0,0.131313], n=(14,13)
r=1000 d=[0] [0] [0] [0]; n=(11,19,17,53)
```