# To-do list

1. Simple example showcasing canonical, fBm and Pearson kernel (and maybe some others? like polynomial and squared exponential). Good to compare a simulation scenario against regularisation and/or GPR. . . . . . . . . . . . . 1
2. Necessary for functional covariates - approximation of the Sobolev-Hilbert space inner product . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7

# Contents

**4 Computational methods for I-priors** **1**

4.1 Toy examples . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1
4.2 Multilevel modelling . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 1
4.3 Longitudinal modelling . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
4.4 Regression with functional covariates . . . . . . . . . . . . . . . . . . . . 6

**Bibliography** **10**

**List of Figures** **11**

**List of Tables** **12**

**List of Theorems** **13**

**List of Definitions** **14**

**List of Symbols** **15**

Haziq Jamil
*Department of Statistics*
*London School of Economics and Political Science*
August 14, 2017

# Chapter 4

# Computational methods for I-priors

## 4.1 Toy examples

Simple example showcasing canonical, fBm and Pearson kernel (and maybe some others? like polynomial and squared exponential). Good to compare a simulation scenario against regularisation and/or GPR.

## 4.2 Multilevel modelling

In this section, a comparison between a standard random effects model and the I-prior approach for estimating varying intercept and varying slopes model is illustrated. We consider a data set which accompanies the MLwiN software on the academic achievements of 4,059 pupils at 65 inner-London schools citeprasbash2012user, R2MLwiN. The response variable of interest are the pupils' (normalised) GCSE scores at age 16 encoded in the variable `normexam`. Also available in the data set is a pupil-specific regressor, which is the London reading test results (`standlrt`) for each pupil taken when they were aged 11.

```
R> data(tutorial, package = "R2MLwiN")
R> str(tutorial[, c("normexam", "school", "standlrt")])
```

```
## 'data.frame': 4059 obs. of  3 variables:
##  $ normexam: num  0.261 0.134 -1.724 0.968 0.544 ...
##  $ school  : Factor w/ 65 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1..
##  $ standlrt: num  0.619 0.206 -1.365 0.206 0.371 ...
```

First, we consider the varying intercept model. A standard approach to fitting this model is the random intercept model, which is based on the assumption that the intercepts are iid normal with zero mean. In R, packages such as **lme4** are able to fit these types of models. In the I-prior approach, the response variable `normexam` is regressed against the covariate `school` indicating the school that pupil had attended, which is assumed to have a nominal effect on GCSE scores. In other words, the regression function lies in the Pearson RKHS. As the variable `school` is a factor-type variable, the **iprior** package knows to treat this variable with the Pearson kernel automatically without user specification.

```
R> # Model 1: Varying intercept model
R> (mod1.fit <- iprior(normexam ~ school, data = tutorial))
```

(output omitted)

```
##
## Call:
## iprior(formula = normexam ~ school, data = tutorial)
##
## RKHS used: Pearson, with a single scale parameter.
##
## Parameter estimates:
##   (Intercept)        lambda           psi
## -0.0001139137  0.0006998747  1.1799071249
```

In Figure **??**(a), the posterior means of the intercepts are plotted for the random effects model and the I-prior model. It can be seen that the estimates are in broad agreement, with conspicuously different estimates for schools 48 (-0.13 vs. -0.38) and 54 (-0.40 vs. -0.58), the I-prior giving the larger estimate in absolute values in both cases. The reason for this is that the I-prior variance for each school's regression function in a Pearson RKHS is inversely proportional to the sample size for that school. A proof of this is given in Appendix **??**. Indeed, schools 48 and 54 have the smallest sample sizes of all schools, namely 2 and 8 respectively, whilst the next smallest is school 37 with 22 students.

Next we consider the varying slope model which regresses, for each school, the GCSE score on the results of the London reading test taken at age 11 (`standlrt`). A standard approach to fitting this model is the random intercept/slopes model, which is based on the assumption that the intercept/slope pairs are iid bivariate normal with zero means. To obtain an I-prior, we assume as above a nominal effect of school, and a linear effect of `standlrt` (using the canonical kernel on this variable). An interaction between the variables `standlrt` and `school` imply that the effect of the covariate `standlrt` varies with each school.

```
R> # Model 2: Varying slope model
R> (mod2.fit <- iprior(normexam ~ school * standlrt, data = tutorial))
```

(output omitted)

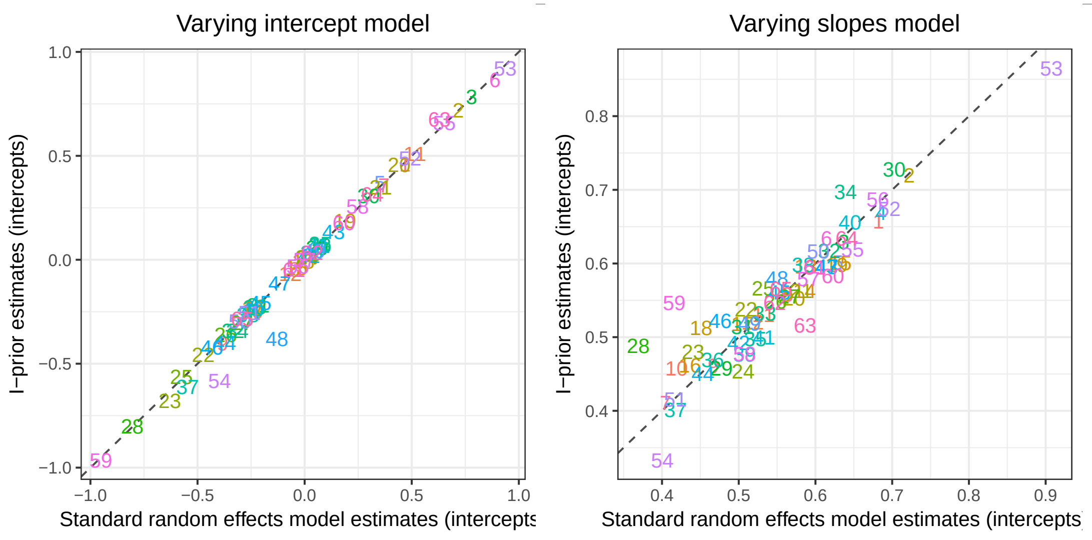

Figure 4.1: Estimated intercepts and slopes for school achievement data under (a) varying intercept (left); and (b) varying slope model. The numbers plotted are the school indices with the identity line for reference.

```
##
## Call:
## iprior(formula = normexam ~ school * standlrt, data = tutorial)
##
## RKHS used: Pearson & Canonical, with multiple scale parameters.
##
## Parameter estimates:
##   (Intercept)       lambda1       lambda2           psi
## -0.0001139137  0.0004234411  0.3731574626  1.8028198235
```

In Figure **??**(b), the posterior means of the slopes obtained using the standard random effects model are plotted against the ones obtained using the I-prior. Again, we see broad agreement of the estimates, but much less so than the varying intercept model.

A limited cross-validation study yielded on average a small advantage of the standard random effects approach in terms of mean squared error, in the order of half a percent, indicating the iid assumption in the random effects models is reasonable. However, an advantage of the I-prior is that no a priori assumption about the distribution of the parameters need to be made. Furthermore, our approach is more parsimonious and allows potentially simpler estimation and testing.

## 4.3 Longitudinal modelling

We consider a balanced longitudinal data set consisting of weights in kilograms of 60 cows, 30 of which were randomly assigned to treatment group A, and the remaining 30 to treatment group B. The animals were weighed 11 times over a 133-day period; the first 10 measurements for each animal were made at two-week intervals and the last measurement was made one week later. This experiment was reported by citekenward1987method, and the data set is included as part of the package **jmcm** in R.

```
R> data(cattle, package = "jmcm")
```

```
## # A tibble: 660 x 4
##        id  time  group weight
##    <fctr> <dbl> <fctr>  <int>
##  1      1     0      A    233
##  2      1    14      A    224
##  3      1    28      A    245
##  4      1    42      A    258
##  5      1    56      A    271
##  6      1    70      A    287
##  7      1    84      A    287
##  8      1    98      A    287
##  9      1   112      A    290
## 10      1   126      A    293
## # ... with 650 more rows
```

The response variable of interest are the `weight` growth curves, and the aim is to investigate whether a treatment effect is present. The usual approach to analyse a longitudinal data set such as this one is to assume that the observed growth curves are realizations of a Gaussian process. For example, citekenward1987method assumed a so-called antedependence structure of order $k$, which assumes an observation depends on the previous $k$ observations, but given these, is independent of any preceeding observations.

Using the I-prior, it is not necessary to assume the growth curves were drawn randomly. Instead, it suffices to assume that they lie in an appropriate function class. For this example, we assume that the function class is the FBM RKHS, i.e., we assume a smooth effect of time on weight. The growth curves form a multidimensional (or functional) response equivalent to a "wide" format of representing repeated measures data. In our analysis using the **iprior** package, we used the "long" format and thus our (unidimensional) sample size $n$ is equal to 60 cows $\times$ 11 repeated measurements. We also have two covariates potentially influencing growth, namely the cow subject `id` and also treatment `group`. The regression model can be thought of as

$$\texttt{weight} = f(\texttt{id}, \texttt{group}, \texttt{time}) + \text{error}.$$

We assume iid errors, and in addition to a smooth effect of `time`, we further assume a nominal effect of both cow `id` and treatment `group` using the Pearson RKHS. In the **iprior** package, factor type objects are treated with the Pearson kernel automatically, and the only `model` option we need to specify is the `kernel = "FBM"` option for the `time` variable. We have opted not to estimate the Hurst coefficient in the interest of computational time, and instead left it at the default value of 0.5. Table 4.1 explains the five models we have fitted.

| Model | Explanation | Formula (`weight ~ ...`) |
|---|---|---|
| 1 | Growth does not vary with treatment nor among cows | `time` |
| 2 | Growth varies among cows only | `id * time` |
| 3 | Growth varies with treatment only | `group * time` |
| 4 | Growth varies with treatment and among cows | `id * time + group * time` |
| 5 | Growth varies with treatment and among cows, with an interaction effect between treatment and cow | `id * group * time` |

Table 4.1: A brief description of the five models fitted using I-priors.

The simplest model fitted was one in which the growth curves do not depend on the treatment effect or individual cows. We then added treatment effect and the cow `id` as covariates, separately first and then together at once. We also assumed that both of these covariates are time-varying, and hence added also the interaction between these covariates and the `time` variable. The final model was one in which an interaction between treatment effect and individual cows was assumed, which varied over time.

All models were first loaded into a `ipriorKernel` object, and then fitted using the `ipriorOptim` function. Compared to the EM algorithm alone, we found that the combination of direct optimization with the EM algorithm in the `ipriorOptim` routine fits the model about six times faster for this data set due to slow convergence of EM algorithm. Here is the code and output for fitting the first model.

```
R> mod1 <- kernL(weight ~ time, data = cattle, model = list(kernel = "FBM"))
R> mod1.fit <- ipriorOptim(mod1)
```

```
## Iteration 0:    Log-likelihood = -40740.339
## Iteration 1:    Log-likelihood = -5416.4625 .......
## Iteration 2:    Log-likelihood = -3347.1579 .......
## Iteration 3:    Log-likelihood = -3096.7290 .......
## EM NOT CONVERGED!
##
```

```
## Now switching to optim...
##
## final  value 2789.600435
## converged
##
## Preparing iprior output... DONE.
```

The `iprior0ptim` routine (see Section **??** for details) performs three EM iterations from a random starting value of the parameters. After the initial EM steps, a direct optimization is carried out using R's built-in optimizer `optim()`. The user has several `control` options to choose from, such as specifying the number of initial EM steps to be performed.

| Model | Formula (`weight ~ ...`) | Log-likelihood | Error S.D. | Number of $\lambda$ parameters |
|---|---|---|---|---|
| 1 | `time` | -2789.60 | 16.22 | 1 |
| 2 | `id * time` | -2792.15 | 16.18 | 2 |
| 3 | `group * time` | -2295.16 | 3.68 | 2 |
| 4 | `id * time + group * time` | -2270.85 | 3.39 | 3 |
| 5 | `id * group * time` | -2250.88 | 3.77 | 3 |

Table 4.2: Summary of the five I-prior models fitted to the cow data set.

The results of the model fit are summarised in Table 4.2. We can test for a treatment effect by testing Model 4 against the alternative that Model 2 is true. The log-likelihood ratio test statistic is $D = -2(-2792.15-(-2270.85)) = 1042.61$ which has an asymptotic chi-squared distribution with $3-2 = 1$ degree of freedom. The $p$-value for this likelihood ratio test is less than $10^{-6}$, so we conclude that Model 4 is significantly better than Model 2.

We can next investigate whether the treatment effect differs among cows by comparing Model 5 against Model 4. As these models have the same number of parameters, we can simply choose the one with the higher likelihood, which is Model 5. We conclude that treatment does indeed have an effect on growth, and that the treatment effect differs among cows. We can use the `plot` function to plot the fitted regression curves onto the cow data set. This is shown in Figure 4.2.

## 4.4 Regression with functional covariates

We illustrate the prediction of a real valued response when one of the covariates is a function using a widely analysed data set for quality control in the food industry. The data[1]contain samples of spectrometric curve of absorbances of 215 pieces of finely

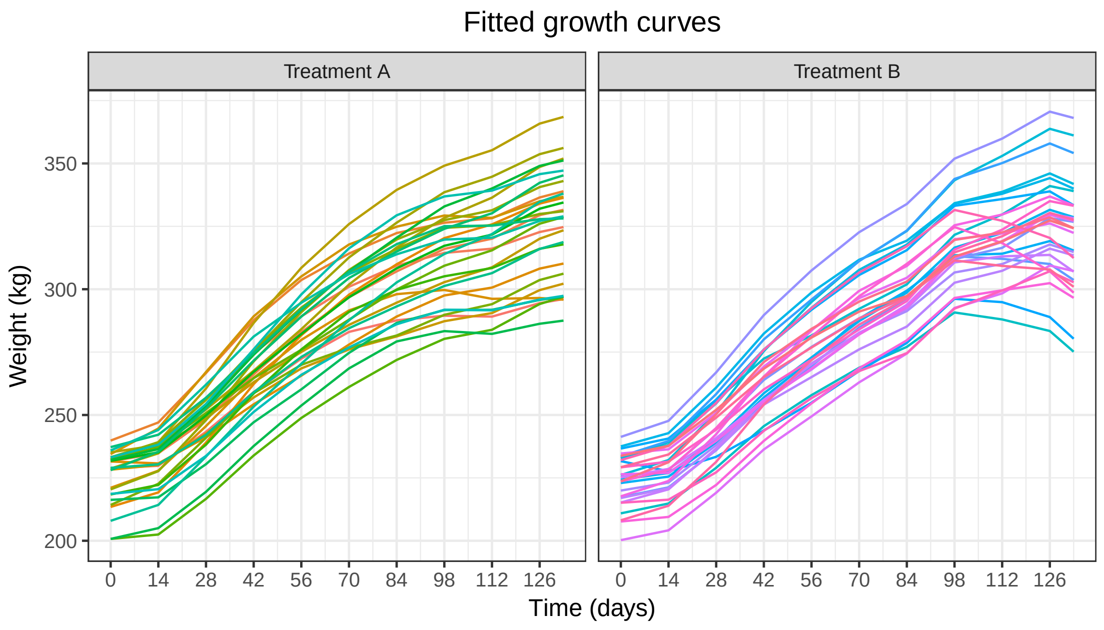

Figure 4.2: A plot of the I-prior fitted regression curves from Model 5. In this model, growth curves differ among cows and by treatment effect (with an interaction between cows and treatment effect), thus producing these 60 individual lines, one for each cow.

chopped meat, along with their water, fat and protein content. These data are recorded on a Tecator Infratec Food and Feed Analyzer working in the wavelength range 850 - 1050 nm by the Near Infrared Transmission (NIT) principle. Absorption data has not been measured continuously, but instead 100 distinct wavelengths were obtained. Figure 4.3 shows a sample of 10 such spectrometric curves.

For our analyses and many others' in the literature, the first 160 observations in the data set are used as a training sample for model fitting, and the remaining 55 observations as a test sample to evaluate the predictive performance of the fitted model. A summary of the various statistical methods applied to this data set, including various I-prior models, can be found in citebergsma2016. The focus here is to use the **iprior** package to fit various I-prior models to the Tecator data set.

Before we began, we preprocessed the spectral curves by taking their first differences . This leaves us with the 99-dimensional covariate, which is saved in the matrix object named `absorpTrain`. Our first modelling attempt is to estimate a linear effect by regressing the responses `fatTrain` against only a single high-dimensional covariate `absorpTrain` using the canonical RKHS. The model is loaded as an `ipriorKernel` object as follows:

2. Necessary for functional covariates - approximation of the Sobolev-Hilbert space inner product

[1]Used with permission from Tecator (see `http://lib.stat.cmu.edu/datasets/tecator` for details). We used the version made available in the dataframe `tecator` from the R package **caret** for our analyses.

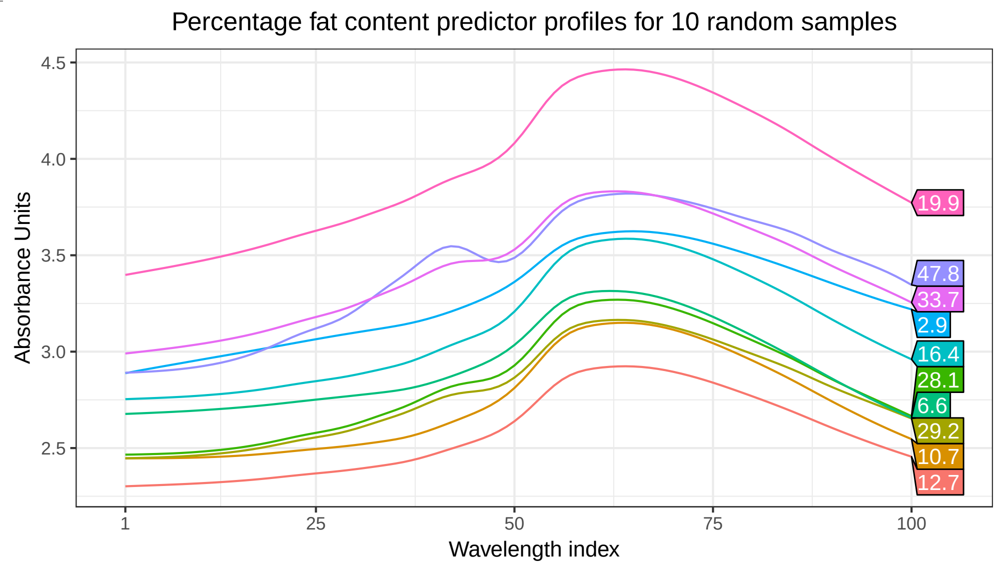

Figure 4.3: Sample of spectrometric curves used to predict fat content of meat. For each meat sample the data consists of a 100 channel spectrum of absorbances and the contents of moisture, fat (numbers shown in boxes) and protein measured in percent. The absorbance is $-\log 10$ of the transmittance measured by the spectrometer. The three contents, measured in percent, are determined by analytic chemistry.

```
R> # Model 1: canonical RKHS (linear)
R> (mod1 <- kernL(y = fatTrain, absorpTrain))

##
## Sample size =  160
## Number of x variables, p =  1
## Number of scale parameters, l =  1
## Number of interactions =  0
##
## Info on H matrix:
##
## List of 1
##  $ absorpTrain: Canonical [1:160, 1:160] 0.000254 0.0003 -0.000231 -..
```

Here, we have used the non-formula syntax because each object after the `y` argument is treated as a single covariate, even if it is multi-dimensional, i.e., a matrix. We could have also used the formula syntax and used the `model` option `one.lam = TRUE`. Note that the canonical RKHS is used by default.

Our second and third model uses a polynomial-type construction of the canonical RKHS, which allows us to add quadratic and cubic terms of the spectral curves. The syntax is as before with the addition of `absorpTrain^b`, which element-wise raises the entries of the matrix `absorpTrain` to the power `b`. To date, the only method to fit these models parsimoniously in **iprior** is by using non-formula syntax with `model` option `order` to control the scale parameters of the RKHS. Both models only have a single parameter. Without specifying the `order` option, additional scale parameters would be fitted, one for each quadratic and cubic term.

```
R> # Model 2: canonical RKHS (quadratic)
R> mod2 <- kernL(y = fatTrain, absorpTrain, absorpTrain ^ 2,
+               model = list(order = c("1", "1^2")))
```

```
R> # Model 3: canonical RKHS (cubic)
R> mod3 <- kernL(y = fatTrain, absorpTrain, absorpTrain ^ 2, absorpTrain ^ 3,
+               model = list(order = c("1", "1^2", "1^3")))
```

Next, we fitted a smooth dependence of fat content on the spectrometric curves using the FBM RKHS. By default, the Hurst coefficient for the FBM RKHS is set to be 0.5. However, we can use the function `fbmOptim()` which is able to compute the maximum likelihood estimate for the Hurst coefficient.

```
R> # Model 4: FBM RKHS (default Hurst = 0.5)
R> mod4 <- kernL(y = fatTrain, absorpTrain, model = list(kernel = "FBM"))
```

Finally, we add an extra covariate (meat moisture content) which is assumed to have a linear effect on fat content. Doing so adds one extra parameter to the model. To specify multiple kernels, we need to include the `model` option `kernel = c("FBM", "Canonical")` to indicate the effect of the respective covariates on the response. This is verified by inspecting the `print` output of the `ipriorKernel` object, and indeed we see that there are now two scale parameters, and the kernel loader correctly assigns the FBM and canonical RKHS to the spectrometric curves and moisture content respectively.

```
R> # Model 5: FBM RKHS + extra covariate
R> (mod5 <- kernL(y = fatTrain, absorpTrain, waterTrain,
+                model = list(kernel = c("FBM", "Canonical"))))

##
## Sample size =  160
## Number of x variables, p =  2
## Number of scale parameters, l =  2
## Number of interactions =  0
##
## Info on H matrix:
##
## List of 2
##  $ absorpTrain: FBM,0.5 [1:160, 1:160] 0.016192 -0.000775 -0.00346 -..
##  $ waterTrain : Canonical [1:160, 1:160] 10.9 58.9 -23.8 -29.7 18.2 ..
```

All of the above models were fitted using `iprior0ptim`, except for the last two model, where we used `fbmOptim` in order to obtain the maximum likelihood estimate for the Hurst coefficient of the FBM RKHS. Predicted values of the test data set can be obtained using the `predict` function

```
R> fatTestPredicted <- predict(mod1.fit, list(absorpTest))
R> head(fatTestPredicted)

## [1] 14.12268 15.85864 15.84706 21.59324 25.22315 26.57978
```

and the root mean squared error (RMSE) calculated for each of the models. It was noted that for some models, different EM starting values gave slightly different results, and we suspect this is a due to numerical issues with the computation of the variance of marginal I-prior distribution. Nonetheless, the predicted values, and hence the RMSE, remain fairly robust.

The results are summarised in Table 4.3. Models 1-3 have the same number of parameters, so a direct comparison can be done, with the model giving the highest likelihood value preferred. In this case, it is the model with a quadratic effect, giving a test RMSE of 1.23. Models with the FBM RKHS gave better prediction still. A smooth effect (Hurst = 0.5) yields a test RMSE of 0.67, and this is improved only slightly by using the maximum likelihood estimate for the Hurst coefficient of 0.519. The best predictive model obtained was the final model, i.e., a smooth effect (Hurst = 0.934) with an additional covariate, giving a test RMSE of 0.68.

| Model | I-prior effect | Log-likelihood | RMSE Train | RMSE Test |
|---|---|---|---|---|
| 1 | Linear | −409.32 | 2.85 | 3.24 |
| 2 | Quadratic | −279.64 | 0.72 | 1.23 |
| 3 | Cubic | −301.26 | 0.99 | 1.65 |
| 4 | Smooth (Hurst = 0.5) | −148.34 | 0.00 | 0.67 |
| 4a | Smooth (Hurst = 0.519) | −146.23 | 0.00 | 0.66 |
| 5 | Smooth (Hurst = 0.934) with additional covariate | −213.51 | 0.31 | 0.54 |

Table 4.3: A summary of the I-prior models fitted on the Tecator data set.

# List of Figures

4.1 Estimated intercepts and slopes for school achievement data under (a) varying intercept (left) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
4.2 A plot of the I-prior fitted regression curves from Model 5 . . . . . . . . . . 7
4.3 Sample of spectrometric curves used to predict fat content of meat . . . . 8

# List of Tables

4.1 A brief description of the five models fitted using I-priors. . . . . . . . . . 5
4.2 Summary of the five I-prior models fitted to the cow data set. . . . . . . . 6
4.3 A summary of the I-prior models fitted on the Tecator data set. . . . . . . 10

# List of Theorems

# List of Definitions

# List of Symbols

| | |
|---|---|
| $\mathrm{N}_p(\mu, \Sigma)$ | $p$-dimensional multivariate normal distribution with mean vector $\mu$ and covariance $\Sigma$. |
| $\sim$ | Is distributed as. |
| $\otimes$ | The tensor product. |

# Index

fractional Brownian motion, *see* fBm

reproducing kernel Hilbert space, *see* RKHS