# Chapter 1

# Automatic Model Construction

> "It would be very nice to have a formal apparatus that gives us some 'optimal' way of recognizing unusual phenomena and inventing new classes of hypotheses that are most likely to contain the true one; but this remains an art for the creative human mind."
>
> – E. T. Jaynes (1985)

In section 1.9, we saw that the choice of kernel determines the type of structure that can be learned by a GP model, and that a wide variety of models could be constructed by adding and multiplying a few base kernels together. However, we did not answer the difficult question of which kernel to use for a given problem. Even for experts, choosing the kernel in GP regression remains something of a black art.

The contribution of this chapter is to show a way to automate the process of building kernels for GP models. We do this by defining an open-ended space of kernels built by adding and multiplying together kernels from a fixed set. We then define a procedure to search over this space to find a kernel which matches the structure in the data.

Searching over such a large, structured model class has two main benefits. First, this procedure has good predictive accuracy, since it tries out a large number of different regression models. Second, this procedure can sometimes discover interpretable structure in datasets. Because GP posteriors can be decomposed (as in **??**), the resulting structures can be examined visually. In section 1.9, we also show how to automatically generate English-language descriptions of the resulting models.

This chapter is based on work done in collaboration with James Robert Lloyd, Roger Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. It was published in Duvenaud et al. (2013) and Lloyd et al. (2014). Myself, James Lloyd and Roger Grosse jointly developed the idea of searching through a grammar-based language of GP models, inspired

by Grosse et al. (2012), and wrote the first versions of the code together. James Lloyd ran most of the experiments, and I constructed most of the figures.

## 1.1 Ingredients of an automatic statistician

Gelman (2013) asks: "How can an artificial intelligence do statistics? ... It needs not just an inference engine, but also a way to construct new models and a way to check models. Currently, those steps are performed by humans, but the AI would have to do it itself". This section will discuss the different parts we think are required to build an artificial intelligence that can do statistics.

1. **An open-ended language of models.** Many learning algorithms consider all models in a class of fixed size. For example, graphical model learning algorithms (Eaton and Murphy, 2007; Friedman and Koller, 2003) search over different connectivity graphs for a given set of nodes. Such methods can be powerful, but human statisticians are sometimes capable of deriving *novel* model classes when required. An automatic search through an open-ended class of models can achieve some of this flexibility, possibly combining existing structures in novel ways.

2. **A search through model space.** Every procedure which eventually considers arbitrarily-complex models must start with relatively simple models before moving on to more complex ones. Thus any search strategy capable of building arbitrarily complex models is likely to resemble an iterative model-building procedure. Just as human researchers iteratively refine their models, search procedures can propose new candidate models based on the results of previous model fits.

3. **A model comparison procedure.** Search strategies requires an objective to optimize. In this work, we use approximate marginal likelihood to compare models, penalizing complexity using the Bayesian Information Criterion as a heuristic. More generally, an automatic statistician needs to somehow check the models it has constructed. Gelman and Shalizi (2012) review the literature on model checking.

4. **A model description procedure.** Part of the value of statistical models comes from helping humans to understand a dataset or a phenomenon. Furthermore, a clear description of the statistical structure found in a dataset helps a user to notice when the dataset has errors, the wrong question was asked, the model-

building procedure failed to capture known structure, a relevant piece of data or constraint is missing, or when a novel statistical structure has been found.

In this chapter, we introduce a system containing simple examples of all the above ingredients. We call this system the automatic Bayesian covariance discovery (ABCD) system. The next four sections of this chapter describe the mechanisms we use to incorporate these four ingredients into a limited example of an artificial intelligence which does statistics.

## 1.2 A language of regression models

As shown in section 1.9, one can construct a wide variety of kernel structures by adding and multiplying a small number of base kernels. We can therefore define a language of GP regression models simply by specifying a language of kernels.

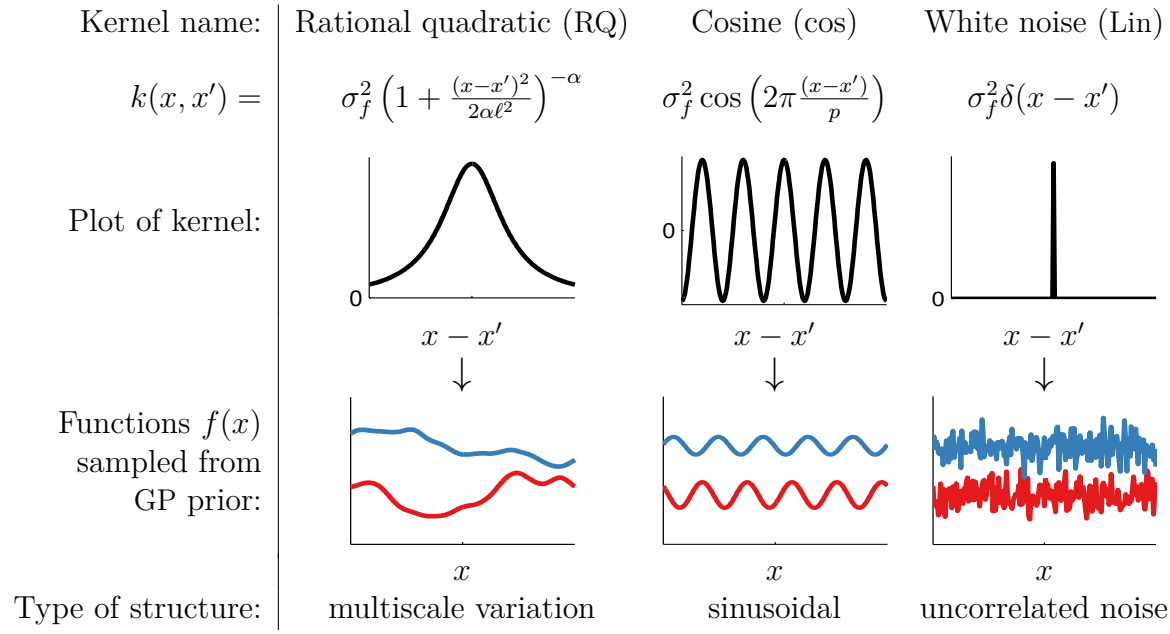


Figure 1.1: New base kernels introduced in this chapter, and the types of structure they encode. Other types of kernels can be constructed by adding and multiplying base kernels together.

Our language of models is specified by a set of base kernels which capture different properties of functions, and a set of rules which combine kernels to yield other valid kernels. In this chapter, we will use such base kernels as white noise (WN), constant

(C), linear (Lin), squared-exponential (SE), rational-quadratic (RQ), sigmoidal ($\boldsymbol{\sigma}$) and periodic (Per). We use a form of Per due to James Lloyd (personal communication) which has its constant component removed, and $\cos(x - x')$ as a special case. Figure 1.1 shows the new kernels introduced in this chapter. For precise definitions of all kernels, see appendix **??**.

To specify an open-ended language of structured kernels, we consider the set of all kernels that can be built by adding and multiplying these base kernels together, which we write in shorthand by:

$$k_1 + k_2 = k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}') \tag{1.1}$$

$$k_1 \times k_2 = k_1(\mathbf{x}, \mathbf{x}') \times k_2(\mathbf{x}, \mathbf{x}') \tag{1.2}$$

The space of kernels constructable by adding and multiplying the above set of kernels contains many existing regression models. Table 1.1 lists some of these, which are discussed in more detail in section 1.7.

| Regression model | Kernel structure | Example of related work |
|---|---|---|
| Linear regression | C + Lin + WN | |
| Polynomial regression | C + $\prod$ Lin + WN | |
| Semi-parametric | Lin + SE + WN | Ruppert et al. (2003) |
| Multiple kernel learning | $\sum$ SE + WN | Gönen and Alpaydın (2011) |
| Fourier decomposition | C + $\sum$ cos + WN | |
| Trend, cyclical, irregular | $\sum$ SE + $\sum$ Per + WN | Lind et al. (2006) |
| Sparse spectrum GPs | $\sum$ cos + WN | Lázaro-Gredilla et al. (2010) |
| Spectral mixture | $\sum$ SE $\times$ cos + WN | Wilson and Adams (2013) |
| Changepoints | e.g. CP(SE, SE) + WN | Garnett et al. (2010) |
| Time-changing variance | e.g. SE + Lin $\times$ WN | |
| Interpretable + flexible | $\sum_d \mathrm{SE}_d + \prod_d \mathrm{SE}_d$ | Plate (1999) |
| Additive GPs | e.g. $\prod_d (1 + \mathrm{SE}_d)$ | Section 1.9 |

Table 1.1: Existing regression models expressible by sums and products of base kernels. $\cos(\cdot, \cdot)$ is a special case of our reparametrized $\mathrm{Per}(\cdot, \cdot)$.

## 1.3 A model search procedure

We explore this open-ended space of regression models using a simple greedy search. At each stage, we choose the highest scoring kernel, and propose modifying it by applying

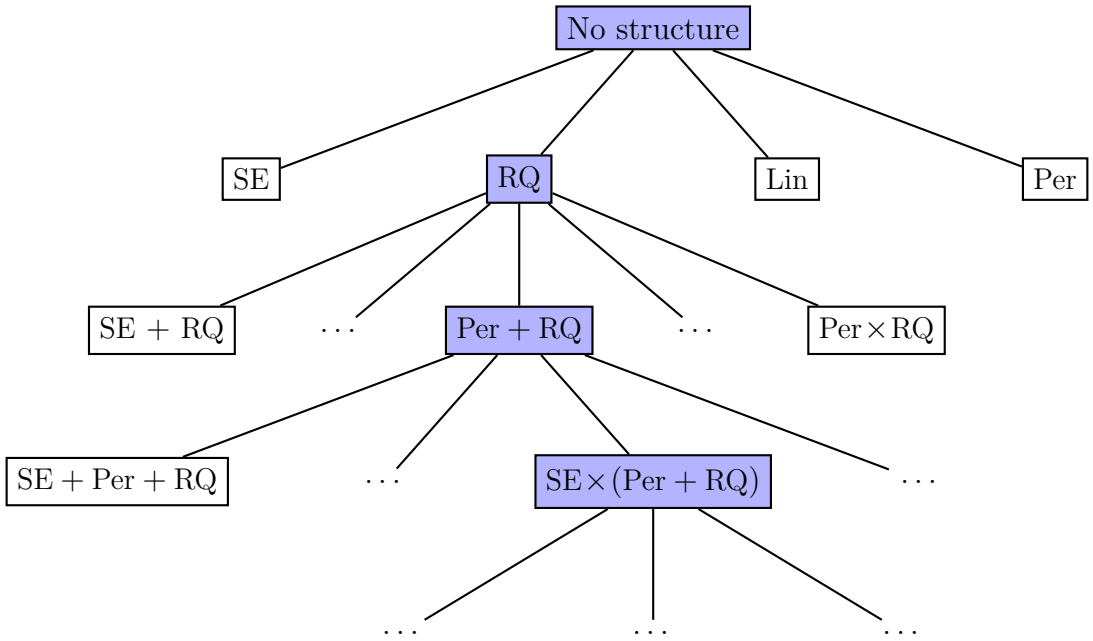


Figure 1.2: An example of a search tree over kernel expressions. Figure 1.3 shows the corresponding model increasing in sophistication as the kernel expression grows.

an operation to one of its parts, that combines or replaces that part with another base kernel. The basic operations we can perform on any part $k$ of a kernel are:

$$\begin{aligned} \text{Replacement:} \quad & k \quad \rightarrow \quad k' \\ \text{Addition:} \quad & k \quad \rightarrow \quad (k + k') \\ \text{Multiplication:} \quad & k \quad \rightarrow \quad (k \times k') \end{aligned}$$

where $k'$ is a new base kernel. These operators can generate all possible algebraic expressions involving addition and multiplication of base kernels. To see this, observe that if we restricted the addition and multiplication rules to only apply to base kernels, we would obtain a grammar which generates the set of algebraic expressions.

Figure 1.2 shows an example search tree followed by our algorithm. Figure 1.3 shows how the resulting model changes as the search is followed. In practice, we also include extra operators which propose commonly-occurring structures, such as changepoints. A complete list is contained in appendix **??**.

Our search operators have rough parallels with strategies used by human researchers to construct regression models. In particular,

- One can look for structure in the residuals of a model, such as periodicity, and then extend the model to capture that structure. This corresponds to adding a new kernel to the existing structure.

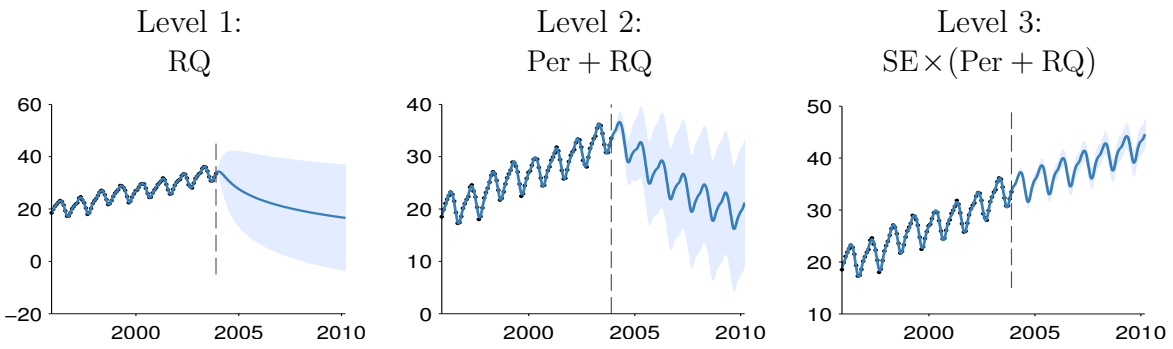


Figure 1.3: Posterior mean and variance for different depths of kernel search on the Mauna Loa dataset, described in section 1.6.1. The dashed line marks the end of the dataset. *Left:* First, the function is only modeled as a locally smooth function, and the extrapolation is poor. *Middle:* A periodic component is added, and the extrapolation improves. *Right:* At depth 3, the kernel can capture most of the relevant structure, and is able to extrapolate reasonably.

- One can start with structure which is assumed to hold globally, such as linearity, but find that it only holds locally. This corresponds to multiplying a kernel structure by a local kernel such as SE.

- One can incorporate input dimensions incrementally, analogous to algorithms like boosting, back-fitting, or forward selection. This corresponds to adding or multiplying with kernels on dimensions not yet included in the model.

### Hyperparameter initialization

Unfortunately, optimizing the marginal likelihood over parameters is not a convex optimization problem, and the space can have many local optima. For example, in data having periodic structure, integer multiples of the true period (harmonics) are often local optima. We take advantage of our search procedure to provide reasonable initializations: all parameters which were part of the previous kernel are initialized to their previous values. Newly introduced parameters are initialized randomly. In the newly proposed kernel, all parameters are then optimized using conjugate gradients. This procedure is not guaranteed to find the global optimum, but it implements the commonly used heuristic of iteratively modeling residuals.

## 1.4 A model comparison procedure

Choosing a kernel requires a method for comparing models. We choose marginal likelihood as our criterion, since it balances the fit and complexity of a model (Rasmussen and Ghahramani, 2001). Conditioned on kernel parameters, the marginal likelihood of a GP can be computed analytically by **??**. In addition, if one compares GP models by the maximum likelihood value obtained after optimizing their kernel parameters, then all else being equal, the model having more free parameters will be chosen. This introduces a bias in favor of more complex models.

We could avoid overfitting by integrating the marginal likelihood over all free parameters, but this integral is difficult to do in general. Instead, we loosely approximate this integral using the Bayesian information criterion (BIC) (Schwarz, 1978):

$$\mathrm{BIC}(M) = \log p(D \mid M) - \frac{1}{2}|M| \log N \tag{1.3}$$

where $p(D|M)$ is the marginal likelihood of the data evaluated at the optimized kernel parameters, $|M|$ is the number of kernel parameters, and $N$ is the number of data points. BIC simply penalizes the marginal likelihood in proportion to how many parameters the model has. Because BIC is a function of the number of parameters in a model, we did not count kernel parameters known to not affect the model. For example, when two kernels are multiplied, one of their output variance parameters becomes redundant, and can be ignored.

The assumptions made by BIC are clearly inappropriate for the model class being considered. For instance, BIC assumes that the data are i.i.d. given the model parameters, which is not true except under a white noise kernel. Other more sophisticated approximations are possible, such as Laplace's approximation. We chose to try BIC first because of its simplicity, and it performed reasonably well in our experiments.

## 1.5 A model description procedure

As discussed in section 1.9, a GP whose kernel is a sum of kernels can be viewed as a sum of functions drawn from different GPs. We can always express any kernel structure as a sum of products of kernels by distributing all products of sums. For example,

$$\mathrm{SE} \times (\mathrm{RQ} + \mathrm{Lin}) = \mathrm{SE} \times \mathrm{RQ} + \mathrm{SE} \times \mathrm{Lin}\,. \tag{1.4}$$

When all kernels in a product apply to the same dimension, we can use the formulas in **??** to visualize the marginal posterior distribution of that component. This decomposition into additive components provides a method of visualizing GP models which disentangles the different types of structure in the model.

The following section shows examples of such decomposition plots. In section 1.9, we will extend this model visualization method to include automatically generated English text explaining types of structure discovered.

## 1.6 Structure discovery in time series

To investigate our method's ability to discover structure, we ran the kernel search on several time-series. In the following example, the search was run to depth 10, using SE, RQ, Lin, Per and WN as base kernels.

### 1.6.1 Mauna Loa atmospheric $CO_2$

First, our method analyzed records of carbon dioxide levels recorded at the Mauna Loa observatory (Tans and Keeling, accessed January 2012). Since this dataset was analyzed in detail by Rasmussen and Williams (2006, chapter 5), we can compare the kernel chosen by our method to a kernel constructed by human experts.

Figure 1.3 shows the posterior mean and variance on this dataset as the search depth increases. While the data can be smoothly interpolated by a model with only a single base kernel, the extrapolations improve dramatically as the increased search depth allows more structure to be included.

Figure 1.4 shows the final model chosen by our method together with its decomposition into additive components. The final model exhibits plausible extrapolation and interpretable components: a long-term trend, annual periodicity, and medium-term deviations. These components have similar structure to the kernel hand-constructed by Rasmussen and Williams (2006, chapter 5):

$$\underbrace{\mathrm{SE}}_{\text{long-term trend}} + \underbrace{\mathrm{SE} \times \mathrm{Per}}_{\text{yearly periodic}} + \underbrace{\mathrm{RQ}}_{\text{medium-term irregularities}} + \underbrace{\mathrm{SE} + \mathrm{WN}}_{\text{short-term noise}} \tag{1.5}$$

We also plot the residuals modeled by a white noise (WN) component, showing that there is little obvious structure left in the data. More generally, some components capture slowly-changing structure while others capture quickly-varying structure, but often there

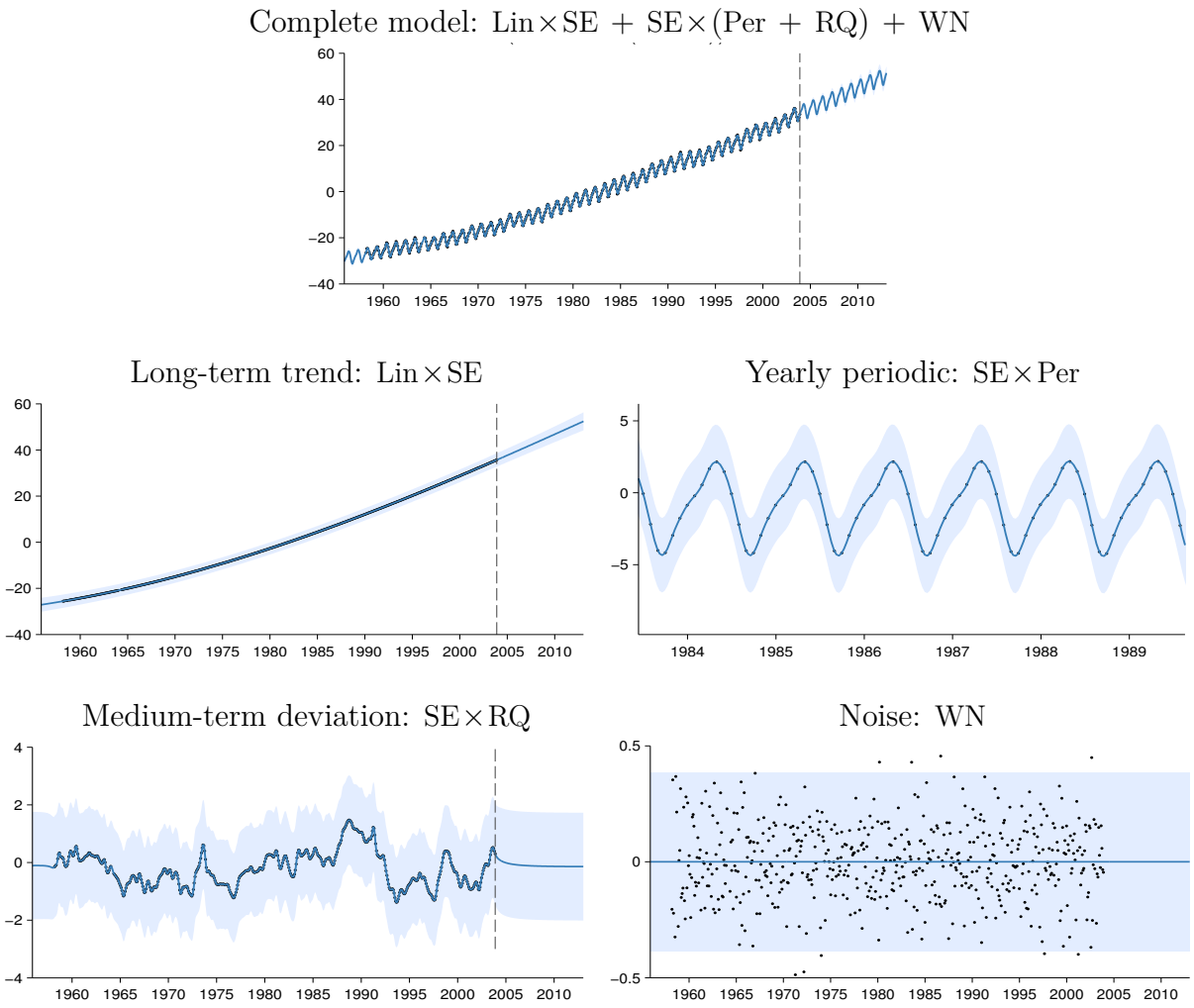


Figure 1.4: *First row:* The full posterior on the Mauna Loa dataset, after a search of depth 10. *Subsequent rows:* The automatic decomposition of the time series. The model is a sum of long-term, yearly periodic, medium-term components, and residual noise, respectively. The yearly periodic component has been rescaled for clarity.

is no hard distinction between "signal" components and "noise" components.

### 1.6.2 Airline passenger counts

Figure 1.5 shows the decomposition produced by applying our method to monthly totals of international airline passengers (Box et al., 1970). We observe similar components to those in the Mauna Loa dataset: a long term trend, annual periodicity, and medium-term deviations. In addition, the composite kernel captures the near-linearity of the long-term trend, and the linearly growing amplitude of the annual oscillations.

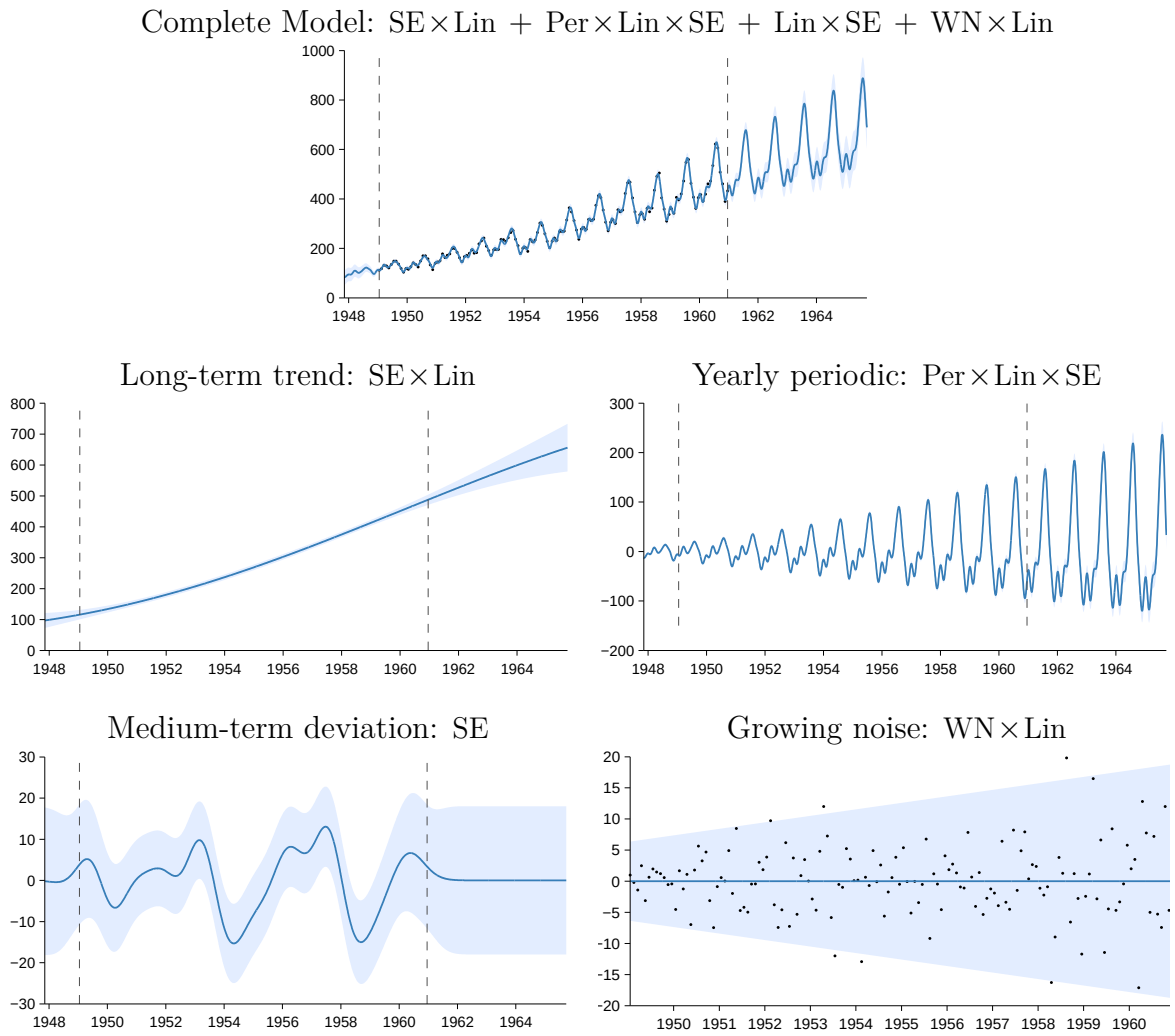


Figure 1.5: *First row:* The airline dataset and posterior after a search of depth 10. *Subsequent rows:* Additive decomposition of posterior into long-term smooth trend, yearly variation, and short-term deviations. Due to the linear kernel, the marginal variance grows over time, making this a heteroskedastic model.

The model search can be run without modification on multi-dimensional datasets (as in section 1.8.4 and **??**), but the resulting structures are more difficult to visualize.

## 1.7 Related work

### Building kernel functions by hand

Rasmussen and Williams (2006, chapter 5) devoted 4 pages to manually constructing

a composite kernel to model the Mauna Loa dataset. Other examples of papers whose main contribution is to manually construct and fit a composite GP kernel are Preotiuc-Pietro and Cohn (2013), Lloyd (2013), and Klenske et al. (2013). These papers show that experts are capable of constructing kernels, in one or two dimensions, of similar complexity to the ones shown in this chapter. However, a more systematic search can consider possibilities that might otherwise be missed. For example, the kernel structure $\mathrm{SE} \times \mathrm{Per} \times \mathrm{Lin}$, while appropriate for the airline dataset, had never been considered by the authors before it was chosen by the automatic search.

### Nonparametric regression in high dimensions

Nonparametric regression methods such as splines, locally-weighted regression, and GP regression are capable of learning arbitrary smooth functions from data. Unfortunately, they suffer from the curse of dimensionality: it is sometimes difficult for these models to generalize well in more than a few dimensions.

Applying nonparametric methods in high-dimensional spaces can require imposing additional structure on the model. One such structure is additivity. Generalized additive models (Hastie and Tibshirani, 1990) assume the regression function is a transformed sum of functions defined on the individual dimensions: $\mathbb{E}[f(\mathbf{x})] = g^{-1}(\sum_{d=1}^{D} f_d(x_d))$. These models have a restricted form, but one which is interpretable and often generalizes well. Models in our grammar can capture similar structure through sums of base kernels along different dimensions, although we have not yet tried incorporating a warping function $g(\cdot)$.

It is possible to extend additive models by adding more flexible interaction terms between dimensions. Section 1.9 considers GP models whose kernel implicitly sums over all possible interactions of input variables. Plate (1999) constructs a special case of this model class, summing an SE kernel along each dimension (for interpretability) plus a single SE-ARD kernel over all dimensions (for flexibility). Both types of model can be expressed in our grammar.

A closely related procedure is smoothing-splines ANOVA (Gu, 2002; Wahba, 1990). This model is a weighted sum of splines along each input dimension, all pairs of dimensions, and possibly higher-dimensional combinations. Because the number of terms to consider grows exponentially with the number of dimensions included in each term, in practice, only one- and two-dimensional terms are usually considered.

Semi-parametric regression (e.g. Ruppert et al., 2003) attempts to combine interpretability with flexibility by building a composite model out of an interpretable, para-

metric part (such as linear regression) and a "catch-all" nonparametric part (such as a GP having an SE kernel). This model class can be represented through kernels such as SE + Lin.

### Kernel learning

There is a large body of work attempting to construct rich kernels through a weighted sum of base kernels, called multiple kernel learning (MKL) (e.g. Bach et al., 2004; Gönen and Alpaydın, 2011). These approaches usually have a convex objective function. However the component kernels, as well as their parameters, must be specified in advance. We compare to a Bayesian variant of MKL in section 1.8, expressed as a restriction of our language of kernels.

Salakhutdinov and Hinton (2008) use a deep neural network with unsupervised pretraining to learn an embedding $g(\mathbf{x})$ onto which a GP with an SE kernel is placed: $\mathrm{Cov}\left[f(\mathbf{x}), f(\mathbf{x}')\right] = k(g(\mathbf{x}), g(\mathbf{x}'))$. This is a flexible approach to kernel learning, but relies mainly on finding structure in the input density $p(\mathbf{x})$. Instead, we focus on domains where most of the interesting structure is in $f(\mathbf{x})$.

Sparse spectrum GPs (Lázaro-Gredilla et al., 2010) approximate the spectral density of a stationary kernel function using sums of Dirac delta functions, which corresponds to kernels of the form $\sum \cos$. Similarly, Wilson and Adams (2013) introduced spectral mixture kernels, which approximate the spectral density using a mixture of Gaussians, corresponding to kernels of the form $\sum \mathrm{SE} \times \cos$. Both groups demonstrated, using Bochner's theorem (Bochner, 1959), that these kernels can approximate any stationary covariance function. Our language of kernels includes both of these kernel classes (see table 1.1).

### Changepoints

There is a wide body of work on changepoint modeling. Adams and MacKay (2007) developed a Bayesian online changepoint detection method which segments time-series into independent parts. This approach was extended by Saatçi et al. (2010) to Gaussian process models. Garnett et al. (2010) developed a family of kernels which modeled changepoints occurring abruptly at a single point. The changepoint kernel (CP) presented in this work is a straightforward extension to smooth changepoints.

### Equation learning

Todorovski and Džeroski (1997), Washio et al. (1999) and Schmidt and Lipson (2009) learned parametric forms of functions, specifying time series or relations between quantities. In contrast, ABCD learns a parametric form for the covariance function, allowing it to model functions which do not have a simple parametric form but still have high-level structure. An examination of the structure discovered by the automatic equation-learning software Eureqa (Schmidt and Lipson, accessed February 2013) on the airline and Mauna Loa datasets can be found in Lloyd et al. (2014).

### Structure discovery through grammars

Kemp and Tenenbaum (2008) learned the structural form of graphs that modeled human similarity judgements. Their grammar on graph structures includes planes, trees, and cylinders. Some of their discrete graph structures have continuous analogues in our language of models. For example, $\mathrm{SE}_1 \times \mathrm{SE}_2$ and $\mathrm{SE}_1 \times \mathrm{Per}_2$ can be seen as mapping the data onto a Euclidean surface and a cylinder, respectively. **??** examined these structures in more detail.

Diosan et al. (2007) and Bing et al. (2010) learned composite kernels for support vector machines and relevance vector machines, respectively, using genetic search algorithms to optimize cross-validation error. Similarly, Kronberger and Kommenda (2013) searched over composite kernels for GPs using genetic programming, optimizing the unpenalized marginal likelihood. These methods explore similar languages of kernels to the one explored in this chapter. It is not clear whether the complex genetic searches used by these methods offer advantages over the straightforward but naïve greedy search used in this chapter. Our search criterion has the advantages of being both differentiable with respect to kernel parameters, and of trading off model fit and complexity automatically. These related works also did not explore the automatic model decomposition, summarization and description made possible by the use of GP models.

Grosse et al. (2012) performed a greedy search over a compositional model class for unsupervised learning, using a grammar of matrix decomposition models, and a greedy search procedure based on held-out predictive likelihood. This model class contains many existing unsupervised models as special cases, and was able to discover diverse forms of structure, such as co-clustering or sparse latent feature models, automatically from data. Our framework takes a similar approach, but in a supervised setting.

Similarly, Steinruecken (2014) showed to automatically perform inference in arbitrary

compositions of discrete sequence models. More generally, Dechter et al. (2013) and Liang et al. (2010) constructed grammars over programs, and automatically searched the resulting spaces.

## 1.8 Experiments

### 1.8.1 Interpretability versus accuracy

BIC trades off model fit and complexity by penalizing the number of parameters in a kernel expression. This can result in ABCD favoring kernel expressions with nested products of sums, producing descriptions involving many additive components after expanding out all terms. While these models typically have good predictive performance, their large number of components can make them less interpretable. We experimented with not allowing parentheses during the search, discouraging nested expressions. This was done by distributing all products immediately after each search operator was applied. We call this procedure ABCD-interpretability, in contrast to the unrestricted version of the search, ABCD-accuracy.

### 1.8.2 Predictive accuracy on time series

We evaluated the performance of the algorithms listed below on 13 real time-series from various domains from the time series data library (Hyndman, accessed July 2013). The pre-processed datasets used in our experiments are available at
`http://github.com/jamesrobertlloyd/gpss-research/tree/master/data/tsdlr`

#### Algorithms

We compare ABCD to equation learning using Eureqa (Schmidt and Lipson, accessed February 2013), as well as six other regression algorithms: linear regression, GP regression with a single SE kernel (squared exponential), a Bayesian variant of multiple kernel learning (MKL) (e.g. Bach et al., 2004; Gönen and Alpaydın, 2011), changepoint modeling (e.g. Fox and Dunson, 2013; Garnett et al., 2010; Saatçi et al., 2010), spectral mixture kernels (Wilson and Adams, 2013) (spectral kernels), and trend-cyclical-irregular models (e.g. Lind et al., 2006).

We set Eureqa's search objective to the default mean-absolute-error. All algorithms besides Eureqa can be expressed as restrictions of our modeling language (see table 1.1),

so we performed inference using the same search and objective function, with appropriate restrictions to the language.

We restricted our experiments to regression algorithms for comparability; we did not include models which regress on previous values of times series, such as auto-regressive or moving-average models (e.g. Box et al., 1970). Constructing a language of autoregressive time-series models would be an interesting area for future research.

### Extrapolation experiments

To test extrapolation, we trained all algorithms on the first 90% of the data, predicted the remaining 10% and then computed the root mean squared error (RMSE). The RMSEs were then standardised by dividing by the smallest RMSE for each data set, so the best performance on each data set has a value of 1.

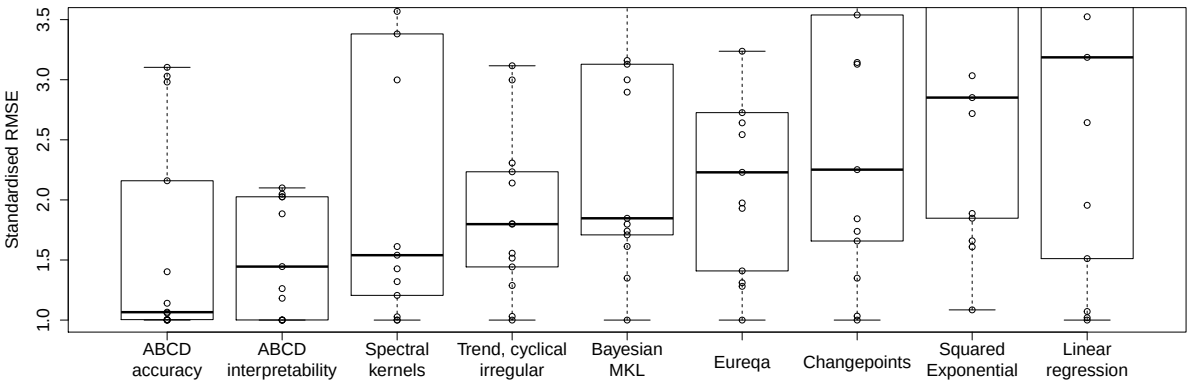


Figure 1.6: Box plot (showing median and quartiles) of standardised extrapolation RMSE (best performance = 1) on 13 time-series. Methods are ordered by median.

Figure 1.6 shows the standardised RMSEs across algorithms. ABCD-accuracy usually outperformed ABCD-interpretability. Both algorithms had lower quartiles than all other methods.

Overall, the model construction methods having richer languages of models performed better: ABCD outperformed trend-cyclical-irregular, which outperformed Bayesian MKL, which outperformed squared-exponential. Despite searching over a rich model class, Eureqa performed relatively poorly. This may be because few datasets are parsimoniously explained by a parametric equation, or because of the limited regularization ability of this procedure.

Not shown on the plot are large outliers for spectral kernels, Eureqa, squared exponential and linear regression with normalized RMSEs of 11, 493, 22 and 29 respectively.

### 1.8.3 Multi-dimensional prediction

ABCD can be applied to multidimensional regression problems without modification. An experimental comparison with other methods can be found in **??**, where it has the best performance on every dataset.

### 1.8.4 Structure recovery on synthetic data

The structure found in the examples above may seem reasonable, but we may wonder to what extent ABCD is consistent – that is, does it recover all the structure in any given dataset? It is difficult to tell from predictive accuracy alone if the search procedure is finding the correct structure, especially in multiple dimensions. To address this question, we tested our method's ability to recover known structure on a set of synthetic datasets.

For several composite kernel expressions, we constructed synthetic data by first sampling 300 locations uniformly at random, then sampling function values at those locations from a GP prior. We then added i.i.d. Gaussian noise to the functions at various signal-to-noise ratios (SNR).

Table 1.2: Kernels chosen by ABCD on synthetic data generated using known kernel structures. $D$ denotes the dimension of the function being modeled. SNR indicates the signal-to-noise ratio. Dashes (–) indicate no structure was found. Each kernel implicitly has a WN kernel added to it.

| True kernel | $D$ | SNR = 10 | SNR = 1 | SNR = 0.1 |
|---|---|---|---|---|
| $\mathrm{SE} + \mathrm{RQ}$ | 1 | SE | $\mathrm{SE} \times \mathrm{Per}$ | SE |
| $\mathrm{Lin} \times \mathrm{Per}$ | 1 | $\mathrm{Lin} \times \mathrm{Per}$ | $\mathrm{Lin} \times \mathrm{Per}$ | SE |
| $\mathrm{SE}_1 + \mathrm{RQ}_2$ | 2 | $\mathrm{SE}_1 + \mathrm{SE}_2$ | $\mathrm{Lin}_1 + \mathrm{SE}_2$ | $\mathrm{Lin}_1$ |
| $\mathrm{SE}_1 + \mathrm{SE}_2 \times \mathrm{Per}_1 + \mathrm{SE}_3$ | 3 | $\mathrm{SE}_1 + \mathrm{SE}_2 \times \mathrm{Per}_1 + \mathrm{SE}_3$ | $\mathrm{SE}_2 \times \mathrm{Per}_1 + \mathrm{SE}_3$ | – |
| $\mathrm{SE}_1 \times \mathrm{SE}_2$ | 4 | $\mathrm{SE}_1 \times \mathrm{SE}_2$ | $\mathrm{Lin}_1 \times \mathrm{SE}_2$ | $\mathrm{Lin}_2$ |
| $\mathrm{SE}_1 \times \mathrm{SE}_2 + \mathrm{SE}_2 \times \mathrm{SE}_3$ | 4 | $\mathrm{SE}_1 \times \mathrm{SE}_2 + \mathrm{SE}_2 \times \mathrm{SE}_3$ | $\mathrm{SE}_1 + \mathrm{SE}_2 \times \mathrm{SE}_3$ | $\mathrm{SE}_1$ |
| $(\mathrm{SE}_1 + \mathrm{SE}_2) \times (\mathrm{SE}_3 + \mathrm{SE}_4)$ | 4 | $(\mathrm{SE}_1 + \mathrm{SE}_2) \times \ldots$ $(\mathrm{SE}_3 \times \mathrm{Lin}_3 \times \mathrm{Lin}_1 + \mathrm{SE}_4)$ | $(\mathrm{SE}_1 + \mathrm{SE}_2) \times \ldots$ $\mathrm{SE}_3 \times \mathrm{SE}_4$ | – |

Table 1.2 shows the results. For the highest signal-to-noise ratio, ABCD usually

recoveres the correct structure. The reported additional linear structure in the last row can be explained the fact that functions sampled from SE kernels with long length-scales occasionally have near-linear trends. As the noise increases, our method generally backs off to simpler structures rather than reporting spurious structure.

#### Source code

All GP parameter optimization was performed by automated calls to the GPML toolbox (Rasmussen and Nickisch, 2010). Source code to perform all experiments is available at `http://www.github.com/jamesrobertlloyd/gp-structure-search`.

## 1.9 Conclusion

This chapter presented a system which constructs a model from an open-ended language, and automatically generates plots decomposing the different types of structure present in the model.

This was done by introducing a space of kernels defined by sums and products of a small number of base kernels. The set of models in this space includes many standard regression models. We proposed a search procedure for this space of kernels, and argued that this search process parallels the process of model-building by statisticians.

We found that the learned structures enable relatively accurate extrapolation in time-series datasets. The learned kernels can yield decompositions of a signal into diverse and interpretable components, enabling model-checking by humans. We hope that this procedure has the potential to make powerful statistical model-building techniques accessible to non-experts.

Some discussion of the limitations of this approach to model-building can be found in **??**, and discussion of this approach relative to other model-building approaches can be found in **??**. The next chapter will show how the model components found by ABCD can be automatically described using English-language text.

# References


Ryan P. Adams and David J.C. MacKay. Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*, 2007. (page 12)

Francis R. Bach, Gert R.G. Lanckriet, and Michael I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the 21st International Conference on Machine learning*, 2004. (pages 12 and 14)

Wu Bing, Zhang Wen-qiong, Chen Ling, and Liang Jia-hong. A GP-based kernel construction and optimization method for RVM. In *International Conference on Computer and Automation Engineering (ICCAE)*, volume 4, pages 419–423, 2010. (page 13)

Salomon Bochner. *Lectures on Fourier integrals*, volume 42. Princeton University Press, 1959. (page 12)

George E.P. Box, Gwilym M. Jenkins, and Gregory C. Reinsel. *Time series analysis: forecasting and control*. John Wiley & Sons, 1970. (pages 9 and 15)

Eyal Dechter, Jon Malmaud, Ryan P. Adams, and Joshua B. Tenenbaum. Bootstrap learning via modular concept discovery. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 1302–1309. AAAI Press, 2013. (page 14)

Laura Diosan, Alexandrina Rogozan, and Jean-Pierre Pecuchet. Evolving kernel functions for SVMs by genetic programming. In *Machine Learning and Applications, 2007*, pages 19–24. IEEE, 2007. (page 13)

David Duvenaud, James Robert Lloyd, Roger B. Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. Structure discovery in nonparametric regression through compositional kernel search. In *Proceedings of the 30th International Conference on Machine Learning*, 2013. (page 1)

Daniel Eaton and Kevin Murphy. Bayesian structure learning using dynamic programming and MCMC. In *Conference on Uncertainty in Artificial Intelligence*, 2007. (page 2)

Emily B. Fox and David B. Dunson. Multiresolution Gaussian processes. In *Advances in Neural Information Processing Systems 25*. MIT Press, 2013. (page 14)

Nir Friedman and Daphne Koller. Being Bayesian about network structure: A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50:95–126, 2003. (page 2)

Roman Garnett, Michael A. Osborne, Steven Reece, Alex Rogers, and Stephen J. Roberts. Sequential Bayesian prediction in the presence of changepoints and faults. *The Computer Journal*, 53(9):1430–1446, 2010. (pages 4, 12, and 14)

Andrew Gelman. Why waste time philosophizing?, 2013. URL `http://andrewgelman.com/2013/02/11/why-waste-time-philosophizing/`. (page 2)

Andrew Gelman and Cosma R. Shalizi. Philosophy and the practice of Bayesian statistics. *British Journal of Mathematical and Statistical Psychology*, 2012. (page 2)

Mehmet Gönen and Ethem Alpaydın. Multiple kernel learning algorithms. *Journal of Machine Learning Research*, 12:2211–2268, 2011. (pages 4, 12, and 14)

Roger B. Grosse, Ruslan Salakhutdinov, William T. Freeman, and Joshua B. Tenenbaum. Exploiting compositionality to explore a large space of model structures. In *Uncertainty in Artificial Intelligence*, 2012. (pages 2 and 13)

Chong Gu. *Smoothing spline ANOVA models*. Springer Verlag, 2002. ISBN 0387953531. (page 11)

Trevor J. Hastie and Robert J. Tibshirani. *Generalized additive models*. Chapman & Hall/CRC, 1990. (page 11)

Rob J. Hyndman. Time series data library, accessed July 2013. URL `http://data.is/TSDLdemo`. (page 14)

Edwin T. Jaynes. Highly informative priors. In *Proceedings of the Second International Meeting on Bayesian Statistics*, 1985. (page 1)

Charles Kemp and Joshua B. Tenenbaum. The discovery of structural form. *Proceedings of the National Academy of Sciences*, 105(31):10687–10692, 2008. (page 13)

Edward D. Klenske, Melanie N. Zeilinger, Bernhard Schölkopf, and Philipp Hennig. Non-parametric dynamics estimation for time periodic systems. In *51st Annual Allerton Conference on Communication, Control, and Computing*, pages 486–493, Oct 2013. (page 11)

Gabriel Kronberger and Michael Kommenda. Evolution of covariance functions for Gaussian process regression using genetic programming. *arXiv preprint arXiv:1305.3794*, 2013. (page 13)

Miguel Lázaro-Gredilla, Joaquin Quiñonero-Candela, Carl E. Rasmussen, and Aníbal R. Figueiras-Vidal. Sparse spectrum Gaussian process regression. *Journal of Machine Learning Research*, 99:1865–1881, 2010. (pages 4 and 12)

Percy Liang, Michael I. Jordan, and Dan Klein. Learning programs: A hierarchical Bayesian approach. In *Proceedings of the 27th International Conference on Machine Learning*, pages 639–646, 2010. (page 14)

Douglas A. Lind, William G. Marchal, and Samuel Adam Wathen. *Basic statistics for business and economics*. McGraw-Hill/Irwin Boston, 2006. (pages 4 and 14)

James Robert Lloyd. GEFCom2012 hierarchical load forecasting: Gradient boosting machines and Gaussian processes. *International Journal of Forecasting*, 2013. (page 11)

James Robert Lloyd, David Duvenaud, Roger B. Grosse, Joshua B. Tenenbaum, and Zoubin Ghahramani. Automatic construction and natural-language description of nonparametric regression models. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2014. (pages 1 and 13)

Tony A. Plate. Accuracy versus interpretability in flexible modeling: Implementing a tradeoff using Gaussian process models. *Behaviormetrika*, 26:29–50, 1999. ISSN 0385-7417. (pages 4 and 11)

Daniel Preotiuc-Pietro and Trevor Cohn. A temporal model of text periodicities using Gaussian processes. In *Conference on Empirical Methods on Natural Language Processing*, pages 977–988. ACL, 2013. (page 11)

Carl E. Rasmussen and Zoubin Ghahramani. Occam's razor. *Advances in Neural Information Processing Systems*, pages 294–300, 2001. (page 7)

Carl E. Rasmussen and Hannes Nickisch. Gaussian processes for machine learning (GPML) toolbox. *Journal of Machine Learning Research*, 11:3011–3015, December 2010. (page 17)

Carl E. Rasmussen and Christopher K.I. Williams. *Gaussian Processes for Machine Learning*, volume 38. The MIT Press, Cambridge, MA, USA, 2006. (pages 8 and 10)

David Ruppert, Matthew P. Wand, and Raymond J. Carroll. *Semiparametric regression*, volume 12. Cambridge University Press, 2003. (pages 4 and 11)

Yunus Saatçi, Ryan D. Turner, and Carl E. Rasmussen. Gaussian process change point models. In *Proceedings of the 27th International Conference on Machine Learning*, pages 927–934, 2010. (pages 12 and 14)

Ruslan Salakhutdinov and Geoffrey Hinton. Using deep belief nets to learn covariance kernels for Gaussian processes. *Advances in Neural information processing systems*, 20:1249–1256, 2008. (page 12)

Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009. ISSN 1095-9203. doi: 10.1126/science.1165893. (page 13)

Michael Schmidt and Hod Lipson. Eureqa [software], accessed February 2013. URL `http://www.eureqa.com`. (pages 13 and 14)

Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2): 461–464, 1978. (page 7)

Christian Steinruecken. *Lossless Data Compression*. PhD thesis, Cavendish Laboratory, University of Cambridge, 2014. (page 13)

Pieter Tans and Ralph Keeling. Monthly mean atmospheric carbon dioxide at the Mauna Loa Observatory, Hawaii, accessed January 2012. URL `www.esrl.noaa.gov/gmd/ccgg/trends/`. (page 8)

Ljupčo Todorovski and Sašo Džeroski. Declarative bias in equation discovery. In *Proceedings of the International Conference on Machine Learning*, pages 376–384, 1997. (page 13)

Grace Wahba. *Spline models for observational data*. Society for Industrial Mathematics, 1990. ISBN 0898712440. (page 11)

Takashi Washio, Hiroshi Motoda, and Yuji Niwa. Discovering admissible model equations from observed data based on scale-types and identity constraints. In *International Joint Conference On Artificial Intelligence*, volume 16, pages 772–779, 1999. (page 13)

Andrew G. Wilson and Ryan P. Adams. Gaussian process kernels for pattern discovery and extrapolation. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1067–1075, 2013. (pages 4, 12, and 14)