Chapter 1

Automatically Building Structured Covariance Functions

"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."

?

In **??**, we saw that the choice of kernel determines the type of structure that can be learnt by a GP model, and that a wide variety of models could be constructed through simply adding and multiplying a few base kernels together. We didn't answer the question, however, of how to tell which kernel to use for a given problem. Even for experts, choosing the kernel in nonparametric regression remains something of a black art.

In this chapter, we'll automate the process of building kernels for GP models. To do so, we need to define an open-ended space of kernels; we'll do this by simply adding and multiplying together simple kernels from a fixed set. We can then simply search over this space to find a kernel which captures as much structure in the data as possible.

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

1.1   Ingredients of an Automatic Statistician

? 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."

In this section, we discuss in more detail the elements we believe are required to build an artificial intelligence that can do statistics.

1. An open-ended language of models   Many statistical procedures consider all models in a class of fixed size - for example, graphical model construction algorithms [1] search over connectivity graphs for a given set of nodes. While these methods can be powerful, human statisticians are capable of deriving novel model classes when required. An automatic search through an open-ended class of models can achieve some of this flexibility, growing the complexity of the model to fit the task at hand, and possibly combining existing structures in novel ways.

2. A Search through model space   An open-ended space of models cannot be searched exhaustively. Just as human researchers iteratively refine their models, search procedures can propose new search directions based on the results of previous model fits. Because any search in an open-ended space must start with relatively simple models before moving on to more complex ones, any model search in an open-ended space will likely resemble a model-building procedure.

3. A Model comparison procedure   An automatic statistician should be able to question the models it has constructed, and formal procedures from model checking provide a way for it to do this. **?** review the literature on model checking. In this work, we use approximate marginal likelihood to compare models, penalizing complexity using the Bayesian Information Criterion as a heuristic.

4. A model description procedure   Part of the value of statistical models comes from enabling 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 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 produce these four ingredients, for this particular example of an artificial intelligence which does statistics.

1.2   A Language of Regression Models

As shown in Chapter **??**, we can construct a wide variety of kernel structures compositionally by adding and multiplying a small number of base kernels. We can therefore define a language of regression models by specifying a language of kernels.

The elements of this language are a set of base kernels capturing different function properties, and a set of composition rules which combine kernels to yield other valid kernels. In this chapter, we'll 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 new parameterization of Per which has $cos(x - x')$ as a special case. For details, see **??**. Table 1.1 shows the new kernels introduced in this chapter.
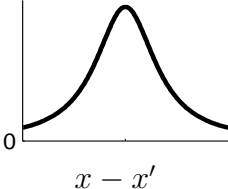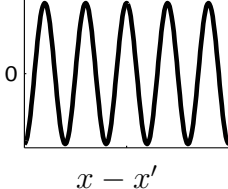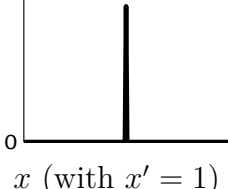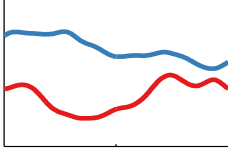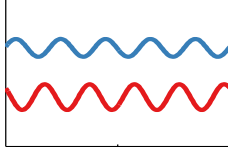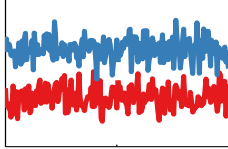
Kernel name:	Rational quadratic (RQ)	Cosine (cos)	White noise (Lin)
$k(x, x') =$	$\exp\left(-\frac{(x-x')^2}{2\ell^2}\right)$	$\cos\left(\frac{2\pi(x-x')}{p}\right)$	$\delta(x - x')$
Plot of kernel:			
	$x - x'$	$x - x'$	x (with $x' = 1$)
	$\downarrow$	$\downarrow$	$\downarrow$
Samples from prior:			
Type of structure:	multiscale variation	sine waves	uncorrelated noise

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

To specify an open-ended language of structured kernels, we'll consider the set of all

kernels that can be built by adding and multiplying these base kernels together:

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

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

Table **??** lists common regression models that can be expressed by this language.

Regression model	Kernel	Related work
Linear regression	$C + \mathrm{Lin} + \mathrm{WN}$	
Kernel ridge regression	$\mathrm{SE} + \mathrm{WN}$	
Linear Semiparametric	$\mathrm{Lin} + \mathrm{SE} + \mathrm{WN}$	(e.g. **?**)
Multiple kernel learning	$\sum \mathrm{SE} + \mathrm{WN}$	(e.g. **?**)
Trend, cyclical, irregular	$\sum \mathrm{SE} + \sum \mathrm{Per} + \mathrm{WN}$	(**?**)
Fourier decomposition	$C + \sum \cos + \mathrm{WN}$	
Sparse spectrum GPs	$\sum \cos + \mathrm{WN}$	(**?**)
Spectral mixture	$\sum \mathrm{SE} \times \cos + \mathrm{WN}$	(**?**)
Changepoints	e.g. $\mathrm{CP}(\mathrm{SE}, \mathrm{SE}) + \mathrm{WN}$	(e.g. **?**)
Heteroscedasticity	e.g. $\mathrm{SE} + \mathrm{Lin} \times \mathrm{WN}$	
Additive + Flexible	$\sum_d \mathrm{SE}_d + \prod_d \mathrm{SE}_d$	(**?**)

Table 1.2 Common regression models expressible by sums and products of base kernels. $\cos(\cdot, \cdot)$ is a special case of our reparametrised $\mathrm{Per}(\cdot, \cdot)$.

1.3   A Model Search Procedure

We explore the space of regression models using a greedy search. At each stage, we choose the highest scoring kernel and expand it by applying all operators to all existing kernels, combining or replacing them with all possible base kernels:

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

These operators can generate all possible algebraic expressions. To see this, observe that if we restricted the $+$ and $\times$ rules to only apply to base kernels, we would obtain a context-free grammar which generates the set of algebraic expressions.

Our search operators are motivated by strategies that human researchers often use to construct kernels. In particular,

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

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

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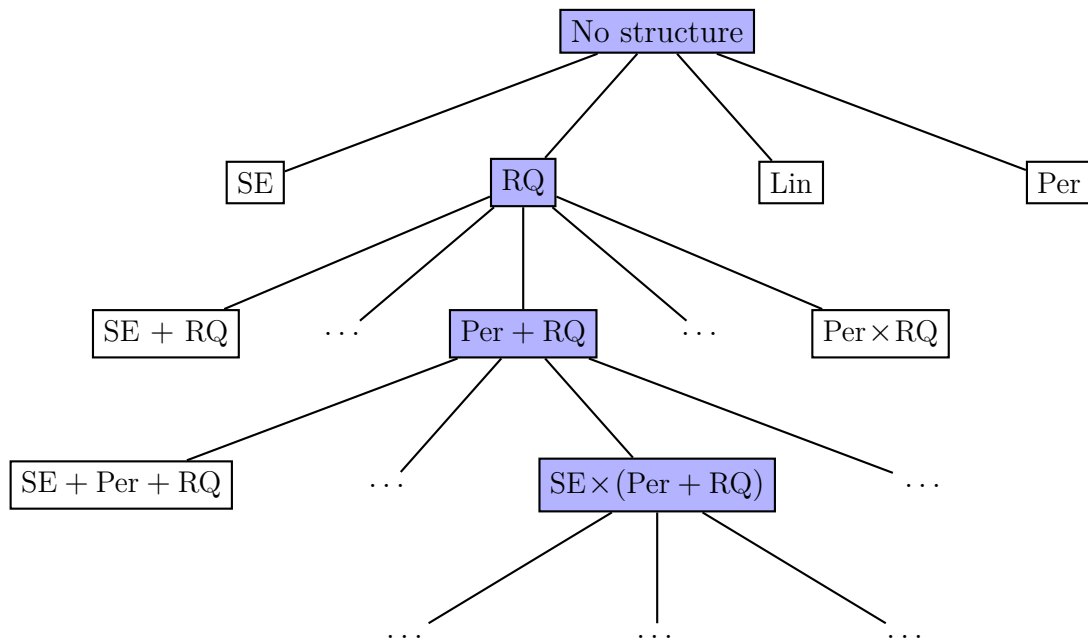

Fig. 1.1 An example of a search tree over kernel expressions. **??** shows the corresponding model increasing in sophistication as the kernel expression grows.

?? shows an example search tree followed by our algorithm. **??** shows how the resulting model changes as the search is followed.

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 with periodic structure, integer multiples of the true period (harmonics) are often local
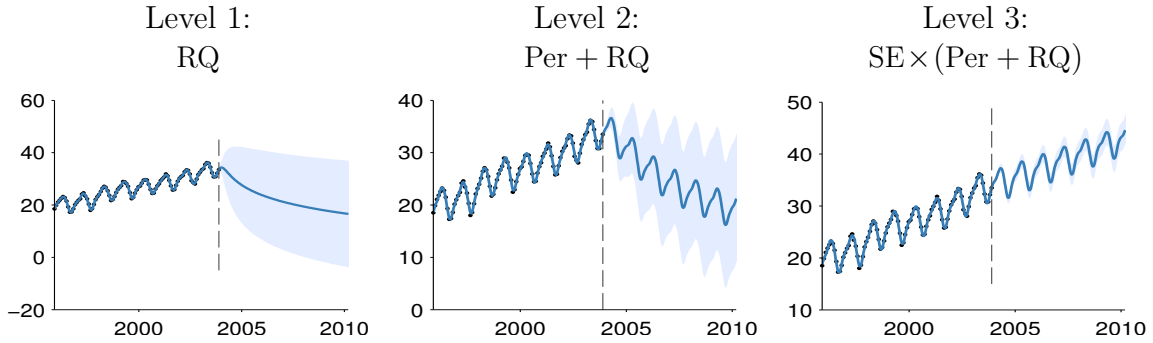
Fig. 1.2 Posterior mean and variance for different depths of kernel search on the Mauna Loa dataset. The dashed line marks the end of the dataset. *Left:* the function is only modeled as a locally smooth function, and the extrapolation is poor. *Middle:* a periodic component is added, and the extrapolation improves. *Right:* at depth 3, the kernel can capture most of the relevant structure, and is able to extrapolate reasonably.

optima. To alleviate this difficulty, we take advantage of our search procedure to provide reasonable initializations: all of the parameters which were part of the previous kernel are initialized to their previous values, while randomly initializing any newly introduced parameters. 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 (**?**). Conditioned on kernel parameters, the marginal likelihood of a GP can be computed analytically. Given a parametric form of a kernel, we can also choose its parameters using marginal likelihood. However, choosing kernel parameters by maximum likelihood raises the possibility of overfitting. In addition, if we compare two classes of kernels by the maximum likelihood attainable over the kernel parameters, then all else being equal, the kernel class having more free parameters will always be chosen.

   We could avoid overfitting by averaging the marginal likelihood over all free parameters, but this integral is hard to do in general. Instead, we approximate this integral

using the Bayesian information criterion (BIC) (**?**):

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

where $p(D|M)$ is the marginal likelihood of the data (given by **??**), $|M|$ 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 adjusted for cases where two parameters were serving the role of one. For example, when two kernels are multiplied, one of their output variance parameters becomes redundant.

Other more sophisticated approximations are possible, such as Laplace's approximation. We chose to first try BIC since it is the simplest thing that could possibly work, and it performed well in our experiments.

1.5 A Model Description Procedure

As discussed in **??**, 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,

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

This decomposition into additive components provides a method of visualizing the learned model, breaking down the different types of structure discovered in the data.

In **??**, we'll extend this model visualization method to include automatically-generated english text explaining the meaning of each type of structure discovered.

1.6 Structure Discovery in Time Series

To investigate our method's ability to discover structure, and to plausibly extrapolate, we ran the kernel search on several time-series. In the following examples, 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

Using our method, we analyzed records of carbon dioxide levels recorded at the Mauna Loa observatory. Since this dataset was analyzed in detail by **?**, we can compare the kernel chosen by our method to a kernel constructed by human experts.

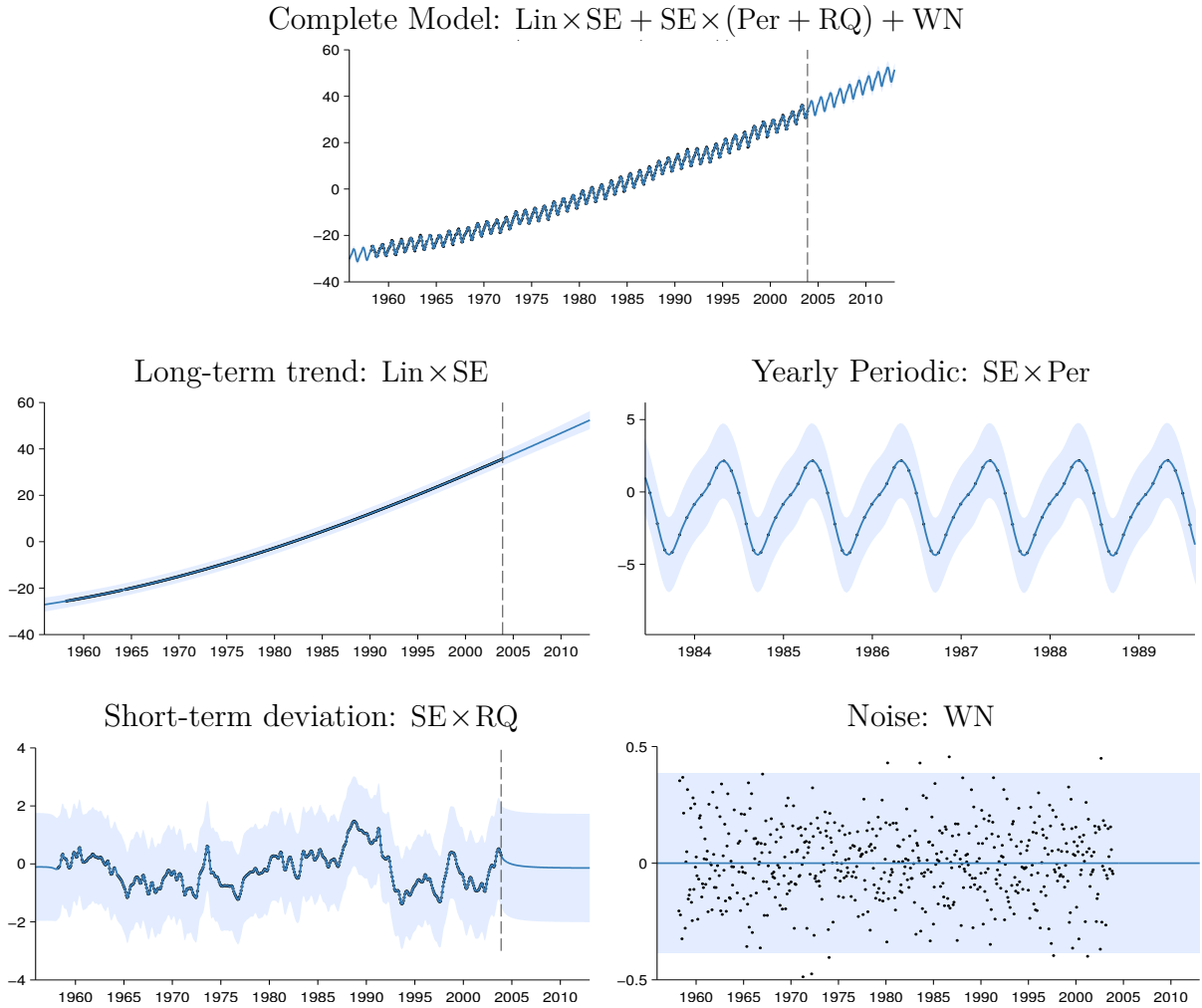Complete Model: $Lin \times SE + SE \times (Per + RQ) + WN$

Fig. 1.3 *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 decompositions contains long-term, yearly periodic, medium-term components, and residual noise, respectively. The yearly periodic component has been rescaled for clarity.

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

Figure **??** shows the final model chosen by our method, together with its decomposition into additive components. The final model exhibits both plausible extrapolation and interpretable components: a long-term trend, annual periodicity and medium-term deviations; the same components chosen by **?**. We also plot the residual noise, showing that there is little obvious structure left in the data.
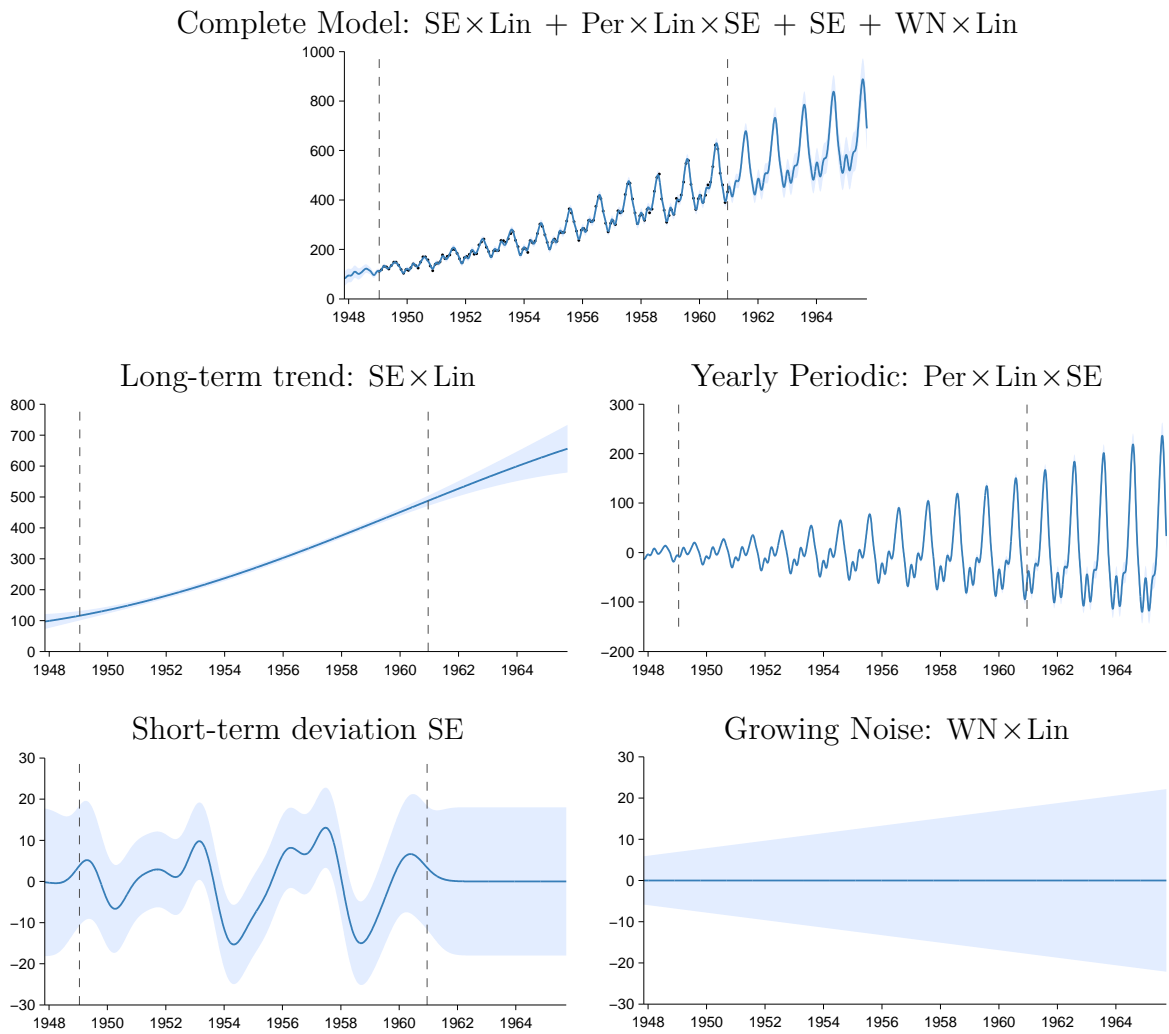
1.6.2 Airline passenger data

Fig. 1.4 *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.

Figure **??** shows the decomposition produced by applying our method to monthly totals of international airline passengers (**?**). We observe similar components to the previous 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.

1.7   Related Work

Building Kernel Functions By Hand

? devote 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 **?** and **?**.

Nonparametric regression in high dimensions

Nonparametric regression methods such as splines, locally weighted regression, and GP regression are popular because they are capable of learning arbitrary smooth functions of the data. Unfortunately, they suffer from the curse of dimensionality: it is very 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 (GAM) 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 limited compositional form, but one which is interpretable and often generalizes well. In our grammar, we can capture such 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 add more flexibility to additive models by considering higher-order interactions between different dimensions. In **??**, we'll consider GP models whose kernel implicitly sums over all possible products of one-dimensional base kernels. **?** constructs a special case of this model class, summing an SE kernel along each dimension, with an SE-ARD kernel (a product of SE over all dimensions). Both of these models can be expressed in our grammar.

A closely related procedure is smoothing-splines ANOVA (**??**). This model is a linear combinations of splines along each dimension, all pairs of dimensions, and possibly higher-order combinations. Because the number of terms to consider grows exponentially

in the order, in practice, only terms of first and second order are usually considered.

Semiparametric regression (e.g. **?**) attempts to combine interpretability with flexibility by building a composite model out of an interpretable, parametric part (such as linear regression) and a 'catch-all' nonparametric part (such as a GP with an SE kernel). This model class can be represented through the kernel SE + Lin.

Kernel learning

There is a large body of work attempting to construct a rich kernel through a weighted sum of base kernels (e.g. **??**). These approaches find the optimal solution in polynomial time, however the component kernels, as well as their parameters, must be specified in advance.

Another approach to kernel learning is to learn an embedding of the data points. **?** learns an embedding of the data into a low-dimensional space, using a fixed kernel structure over that space. This model is typically used in unsupervised tasks and requires an expensive integration or optimisation over potential embeddings when generalizing to test points.

? use a deep neural network to learn an embedding; this is a flexible approach to kernel learning but relies upon 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 (**?**) approximate the spectral density of a stationary kernel function using delta functions which corresponds to kernels of the form $\sum \cos$. Similarly, **?** introduce spectral mixture kernels which approximate the spectral density using a scale-location mixture of Gaussian distributions corresponding to kernels of the form $\sum \text{SE} \times \cos$. Both demonstrate, using Bochner's theorem (**?**), that these kernels can approximate any stationary covariance function. Our language of kernels includes both of these kernel classes (see **??**).

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. **?**). These approaches find the optimal solution in polynomial time, but only if the component kernels and parameters are pre-specified. We compare to a Bayesian variant of MKL in **??** which is expressed as a restriction of our language of kernels.

Equation learning

?, **?** and **?** learn parametric forms of functions specifying time series, or relations

between quantities. In contrast, ABCD learns a parametric form for the covariance, allowing it to model functions without a simple parametric form. A examination of the structure discovered by the automatic equation-learning software Eureqa (**?**) on the airline and Mauna Loa datasets can be found in **??**.

Structure Discovery Through Grammars

? learned the structural form of a graph used to model human similarity judgments. Examples of graphs included planes, trees, and cylinders. Some of their discrete graph structures have continous analogues in our own space. For example, $SE_1 \times SE_2$ and $SE_1 \times Per_2$ can be seen as mapping the data to a plane and a cylinder, respectively.

? and **?** learn composite kernels for support vector machines and relevance vector machines respectively, using genetic search algorithms to optimize cross-validation error. Similarly, **?** search over composite kernels for GPs using genetic programming, optimizing the unpenalized marginal likelihood. These methods explore similar languages of kernels to that 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 work employs a search criterion which is both differentiable with respect to kernel parameters, and also trades off model fit an complexity. This prior work also did not explore the automatic model decomposition, summarization and description made possible by the use of GP models.

? 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 likelihood. This model class contains many existing unsupervised models as special cases and was able to discover such structure automatically from data. Our framework takes a similar approach, but in a supervised setting.

Similarly, **?** showed to automatically perform inference in arbitrary compositions of discrete sequence models. More generally, **?** and **?** 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 when expressions are expanded. While these models typically have good predictive performance, the large number of components can make them less interpretable. We experimented with not allowing parentheses during the search, discouraging nested expressions. 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 evaluate the performance of the algorithms listed below on 13 real time-series from various domains from the time series data library (**?**).

Algorithms

We compare ABCD to equation learning using Eureqa (**?**), 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. **?**), change point modeling (e.g. **???**), spectral mixture kernels (**?**) (spectral kernels) and trend-cyclical-irregular models (e.g. **?**).

We use the default mean absolute error criterion when using Eureqa. All other algorithms can be expressed as restrictions of our modeling language (see table **??**) so we perform inference using the same search methodology and selection criterion[1] with appropriate restrictions to the language. For MKL, trend-cyclical-irregular, and spectral kernels, the greedy search procedure of ABCD corresponds to a forward-selection algorithm. For squared-exponential and linear regression, the procedure corresponds to standard marginal likelihood optimization. More advanced inference methods are typically used for changepoint modeling, but we use the same inference method for all algorithms for comparability.

We restricted to regression algorithms for comparability; this excludes models which regress on previous values of times series, such as autoregressive or moving-average models (e.g. **?**). Constructing a language of autoregression time-series models would be an interesting area for future research.

Extrapolation

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

are then standardised by dividing by the smallest RMSE for each data set so that the best performance on each data set will have a value of 1.
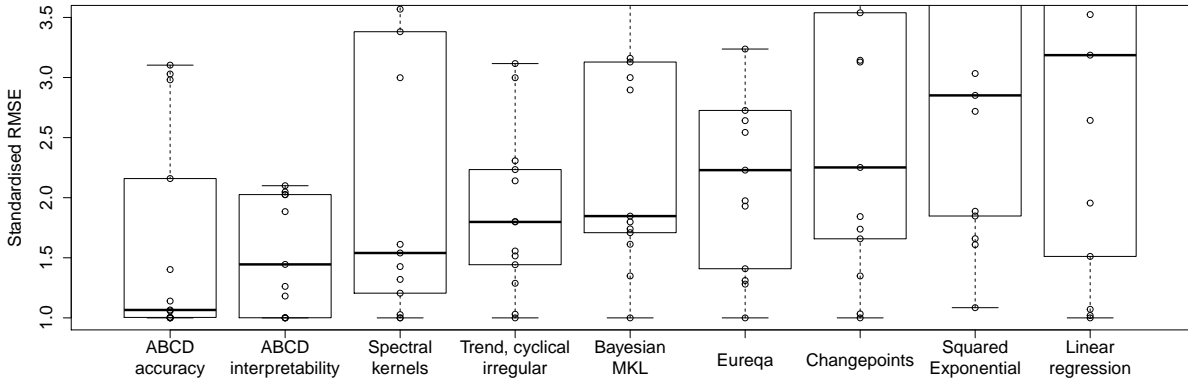

Fig. 1.5 Box plot (showing median and quartiles) of standardised extrapolation RMSE (best performance = 1) on 13 time-series. The methods are ordered by median.

Figure **??** shows the standardised RMSEs across algorithms. ABCD-accuracy outperforms ABCD-interpretability. Both algorithms have lower quartiles than all other methods.

Overall, the model construction methods with greater capacity perform better: ABCD outperforms trend-cyclical-irregular, which outperforms Bayesian MKL, which outperforms squared-exponential. Despite searching over a rich model class, Eureqa performs relatively poorly, since very few datasets are parsimoniously explained by a parametric equation.

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

Interpolation

To test the ability of the methods to interpolate, we randomly divided each data set into equal amounts of training data and testing data. The results are similar to those for extrapolation, and are included in the appendix.

1.8.3 High-dimensional Prediction

ABCD can also be applied to multidimensional regression problems without modification. An experimental comparison with other methods can be found in **??**, where it

outperforms a wide variety of multidimensional regression methods.

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, when do we recover all the structure in a dataset? Especially in multiple dimensions, it is hard to tell from predictive accuracy alone if the search procedure is finding the correct structure. 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 points uniformly at random, then sampling function values at those points from a GP prior. We then added i.i.d. Gaussian noise to the functions, at various signal-to-noise ratios (SNR).

Table 1.3 Kernels chosen by our method on synthetic data generated using known kernel structures. D denotes the dimension of the functions being modeled. SNR indicates the signal-to-noise ratio. Dashes - indicate no structure was found. Each kernel implicity has a WN kernel included.

True Kernel	D	SNR $= 10$	SNR $= 1$	SNR $= 0.1$
SE $+$ RQ	1	SE	SE $\times$ Per	SE
Lin $\times$ Per	1	Lin $\times$ Per	Lin $\times$ Per	SE
SE$_1$ $+$ RQ$_2$	2	SE$_1$ $+$ SE$_2$	Lin$_1$ $+$ SE$_2$	Lin$_1$
SE$_1$ $+$ SE$_2$ $\times$ Per$_1$ $+$ SE$_3$	3	SE$_1$ $+$ SE$_2$ $\times$ Per$_1$ $+$ SE$_3$	SE$_2$ $\times$ Per$_1$ $+$ SE$_3$	-
SE$_1$ $\times$ SE$_2$	4	SE$_1$ $\times$ SE$_2$	Lin$_1$ $\times$ SE$_2$	Lin$_2$
SE$_1$ $\times$ SE$_2$ $+$ SE$_2$ $\times$ SE$_3$	4	SE$_1$ $\times$ SE$_2$ $+$ SE$_2$ $\times$ SE$_3$	SE$_1$ $+$ SE$_2$ $\times$ SE$_3$	SE$_1$
(SE$_1$ $+$ SE$_2$) $\times$ (SE$_3$ $+$ SE$_4$)	4	(SE$_1$ $+$ SE$_2$)$\times \ldots$ (SE$_3$ $\times$ Lin$_3$ $\times$ Lin$_1$ $+$ SE$_4$)	(SE$_1$ $+$ SE$_2$)$\times \ldots$ SE$_3$ $\times$ SE$_4$	-

Table **??** shows the results. For the highest SNR, the method finds all relevant structure in all but one case. The reported additional linear structure is explainable by 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 over-fitting.

Source Code

Source code to perform all experiments is available on at `www.github.com/jamesrobertlloyd/gpss-research`. All GP parameter optimisation was performed by automated calls to

the GPML toolbox, available at `www.gaussianprocess.org/gpml/code/`.

1.9   Discussion

Towards the goal of automating statistical modeling, we developed a system which constructs an appropriate model from an open-ended language, and automatically generates plots decomposing the different types of structure present in the model.

We acheived this by introducing a space of composite kernels defined compositionally as sums and products of a small number of base kernels. The set of models included 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 scientific discovery, and of model-building by statisticians.

We found that the learned structures are often capable of accurate extrapolation in complex time-series datasets, and are competitive with widely used kernel classes and kernel combination methods on a variety of prediction tasks. The learned kernels often 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.

In the next chapter, we'll see how the model components found by this procedure can be automatically described in terms of english-language text.

1.10   Open Questions

How well will the search work for classification?

While we focus on Gaussian process regression, we believe our kernel search method can be extended to other supervised learning frameworks such as classification or ordinal regression, or to other kinds of kernel architectures such as kernel SVMs. However, it might be the case that it is much more difficult to discover structure from class labels alone, since they usually contain less information about the underlying process than does regression data.

How can the search be made faster?

It is clear that the search procedure used in this chapter leaves much room for improvement. Presumably, some sort of model-based search would be worthwhile, since

evaluating each kernel expression is relatively expensive. However, this raises the question of how to build an open-ended model of the relative marginal likelihood of GP models. In order to use a GP for this task, one would need to create a kernel over kernels.

One interesting possibility is that the model of model performance could generate its own data offline, by generating synthetic regression datasets, then scoring many kernel structures on those datasets in order to improve its model of kernel performance.

Can we build similar languages in other model classes?

The compositionality of diverse types of kernels is what allowed a rich language of GP models. Can we build similar languages of other model classes?

As noted in **??**, languages have already been constructed on several powerful model classes: matrix decompositions, sequence models, graphical models, and functional programs. Building languages of models that support efficient search as well and interpretable decompositions could be a fruitful direction.

References

F. Bach. Exploring large feature spaces with hierarchical multiple kernel learning. In *Advances in Neural Information Processing Systems*, pages 105–112. 2009. (page 11)

Francis R Bach, Gert RG Lanckriet, and Michael I Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the twenty-first international conference on Machine learning*, page 6. ACM, 2004. (pages 4, 11, and 13)

W. Bing, Z. Wen-qiong, C. Ling, and L. 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 12)

Salomon Bochner. *Lectures on Fourier integrals*, volume 42. Princeton University Press, 1959. (page 11)

George EP Box, Gwilym M Jenkins, and Gregory C Reinsel. *Time series analysis: forecasting and control*. Wiley. com, 2013. (page 14)

G.E.P. Box, G.M. Jenkins, and G.C. Reinsel. *Time series analysis: forecasting and control*. 1976. (page 9)

M. Christoudias, R. Urtasun, and T. Darrell. Bayesian localized multiple kernel learning. *Technical report, EECS Department, University of California, Berkeley*, 2009. (page 11)

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 12)

L. Diosan, A. Rogozan, and J.P. Pecuchet. Evolving kernel functions for SVMs by genetic programming. In *Machine Learning and Applications, 2007*, pages 19–24. IEEE, 2007. (page 12)

E.B. Fox and D.B. Dunson. Multiresolution Gaussian Processes. In *Neural Information Processing Systems 25*. MIT Press, 2013. (page 13)

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 and 13)

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

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

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. (page 12)

C. Gu. *Smoothing spline ANOVA models*. Springer Verlag, 2002. ISBN 0387953531. (page 10)

Rob J. Hyndman. Time series data library, Accessed summer 2013. URL http://data.is/TSDLdemo. (page 13)

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

C. Kemp and J.B. Tenenbaum. The discovery of structural form. *Proceedings of the National Academy of Sciences*, 105(31):10687–10692, 2008. (page 12)

E.D. Klenske, M.N. Zeilinger, B. Scholkopf, and P. Hennig. Nonparametric dynamics estimation for time periodic systems. In *Communication, Control, and Computing (Allerton), 2013 51st Annual Allerton Conference on*, pages 486–493, Oct 2013. (page 9)

Gabriel Kronberger and Michael Kommenda. Evolution of covariance functions for gaussian process regression using genetic programming. *arXiv preprint arXiv:1305.3794*, 2013. (page 12)

N. Lawrence. Probabilistic non-linear principal component analysis with gaussian process latent variable models. *The Journal of Machine Learning Research*, 6:1783–1816, 2005. (page 11)

Miguel Lázaro-Gredilla, Joaquin Quiñonero-Candela, Carl Edward Rasmussen, and Aníbal R Figueiras-Vidal. Sparse spectrum gaussian process regression. *The Journal of Machine Learning Research*, 99:1865–1881, 2010. (pages 4 and 11)

Percy Liang, Michael I Jordan, and Dan Klein. Learning programs: A hierarchical bayesian approach. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 639–646, 2010. (page 12)

Douglas A Lind, William G Marchal, Samuel Adam Wathen, and Business Week Magazine. *Basic statistics for business and economics.* McGraw-Hill/Irwin Boston, 2006. (pages 4 and 13)

James Robert Lloyd. GEFCom2012 hierarchical load forecasting: Gradient boosting machines and gaussian processes. *International Journal of Forecasting*, 2013. (page 9)

Nutonian. Eureqa, 2011. URL http://www.nutonian.com/. (pages 12 and 13)

T.A. Plate. Accuracy versus interpretability in flexible modeling: Implementing a trade-off using Gaussian process models. *Behaviormetrika*, 26:29–50, 1999. ISSN 0385-7417. (pages 4 and 10)

Carl Edward Rasmussen and Zoubin Ghahramani. Occam's razor. *Advances in neural information processing systems*, pages 294–300, 2001. (page 6)

C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning*, volume 38. The MIT Press, Cambridge, MA, USA, 2006. (pages 8 and 9)

D. Ruppert, M.P. Wand, and R.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 (ICML-10)*, pages 927–934, 2010. (page 13)

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 11)

Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, April 2009. ISSN 1095-9203. doi: 10.1126/science. 1165893. (page 12)

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

Christian Steinruecken. *Bayesian Compression*. PhD thesis, University of Cambridge, 2014. (page 12)

L. Todorovski and S. Dzeroski. Declarative bias in equation discovery. In *International Conference on Machine Learning*, pages 376–384, 1997. (page 12)

G. Wahba. *Spline models for observational data*. Society for Industrial Mathematics, 1990. ISBN 0898712440. (page 10)

T. Washio, H. Motoda, Y. Niwa, et al. Discovering admissible model equations from observed data based on scale-types and identity constraints. In *International Joint Conference On Artifical Intelligence*, volume 16, pages 772–779, 1999. (page 12)

Andrew Gordon Wilson and Ryan Prescott Adams. Gaussian process covariance kernels for pattern discovery and extrapolation. *arXiv: 1302.4245*, June 2013. (pages 4, 11, and 13)