The Automatic Statistician

Zoubin Ghahramani

Department of Engineering
University of Cambridge, UK

zoubin@eng.cam.ac.uk
http://learning.eng.cam.ac.uk/zoubin/

James Lloyd, David Duvenaud (Cambridge)
and Roger Grosse, Josh Tenenbaum (MIT)
Imperial College Lectures, 2014

Motivation

- We live in an era of **abundant data**
- Scientific, commercial and societal uses of this data drive the need for exploratory data analysis and prediction methods, but there are **too few experts statisticians and data scientists** to provide these services.
- Many aspects of statistical inference can be automated, and one of the goals of machine learning and artificial intelligence is to develop powerful tools for understanding data that require minimal expert input.
- By trying to build an "Automatic Statistician" we can
  - provide a set of useful tools for understanding certain kinds of data
  - uncover challenging research problems in automated inference, model construction and comparison, and data visualisation and interpretation
  - advance the field of machine learning in general

Ingredients

- Probabilistic modelling
- Model selection and marginal likelihoods
- Bayesian nonparametrics
- Gaussian processes
- Change-point kernels

Probabilistic Modelling

- A model describes data that one could observe from a system
- If we use the mathematics of probability theory to express all forms of uncertainty and noise associated with our model...
- ...then *inverse probability* (i.e. Bayes rule) allows us to infer unknown quantities, adapt our models, make predictions and learn from data.

Bayesian Machine Learning

Everything follows from two simple rules:

Sum rule: $P(x) = \sum_y P(x, y)$

Product rule: $P(x, y) = P(x)P(y|x)$

$$P(\theta|\mathcal{D}, m) = \frac{P(\mathcal{D}|\theta, m)P(\theta|m)}{P(\mathcal{D}|m)}$$

$P(\mathcal{D}\|\theta, m)$	likelihood of parameters θ in model m
$P(\theta\|m)$	prior probability of θ
$P(\theta\|\mathcal{D}, m)$	posterior of θ given data $\mathcal{D}$

Prediction:

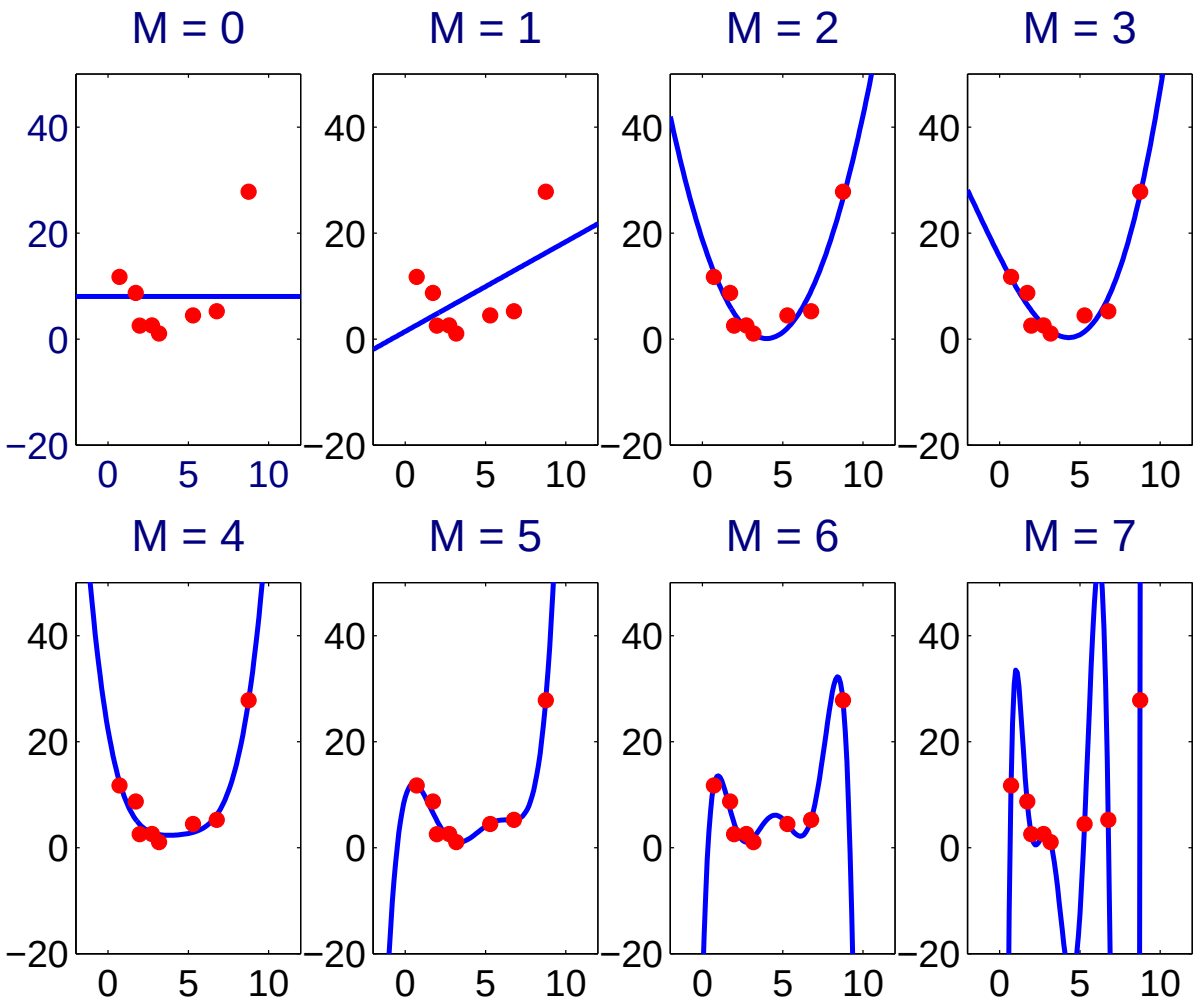
$$P(x|\mathcal{D}, m) = \int P(x|\theta, \mathcal{D}, m)P(\theta|\mathcal{D}, m)d\theta$$

Model Comparison:

$$P(m|\mathcal{D}) = \frac{P(\mathcal{D}|m)P(m)}{P(\mathcal{D})}$$

$$P(\mathcal{D}|m) = \int P(\mathcal{D}|\theta, m)P(\theta|m)\, d\theta$$

Model Comparison



Bayesian Occam's Razor and Model Comparison

Compare model classes, e.g. m and m', using posterior probabilities given $\mathcal{D}$:

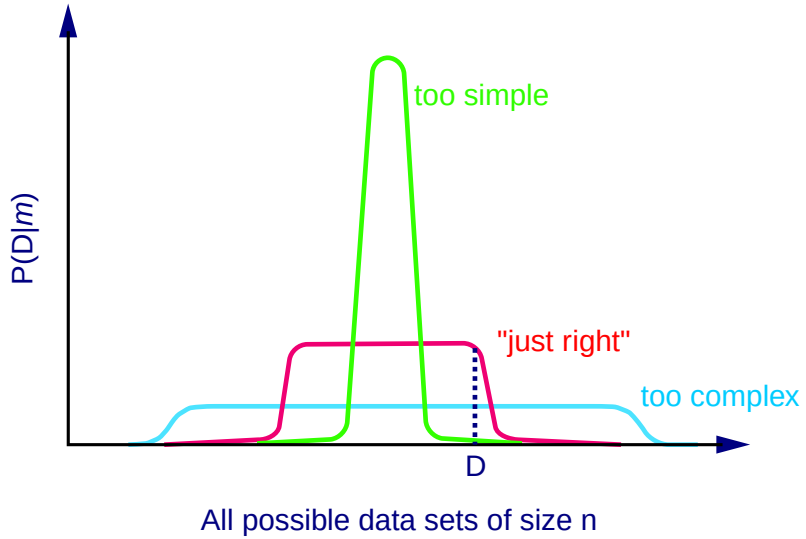
$$p(m|\mathcal{D}) = \frac{p(\mathcal{D}|m)\,p(m)}{p(\mathcal{D})}, \qquad p(\mathcal{D}|m) = \int p(\mathcal{D}|\boldsymbol{\theta}, m)\; p(\boldsymbol{\theta}|m)\; d\boldsymbol{\theta}$$

Interpretations of the Marginal Likelihood ("model evidence"):

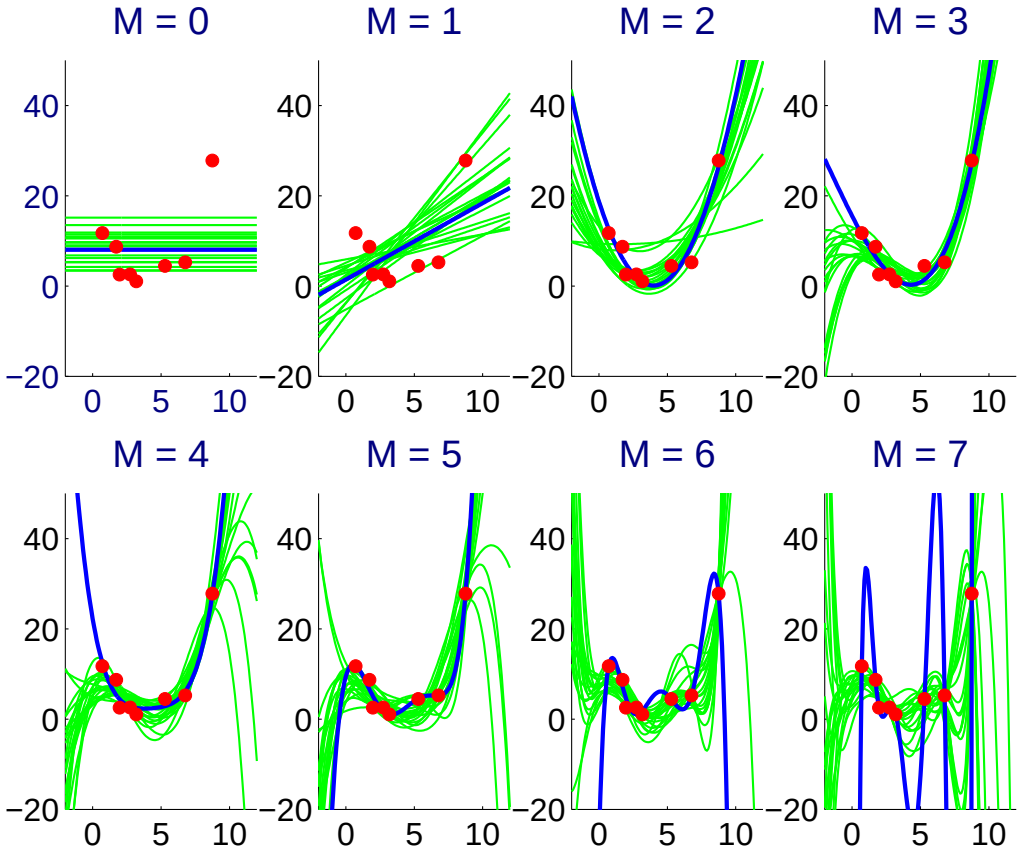
- The probability that *randomly selected* parameters from the prior would generate $\mathcal{D}$.
- Probability of the data under the model, *averaging* over all possible parameter values.
- $\log_2\left(\frac{1}{p(\mathcal{D}|m)}\right)$ is the number of *bits of surprise* at observing data $\mathcal{D}$ under model m.

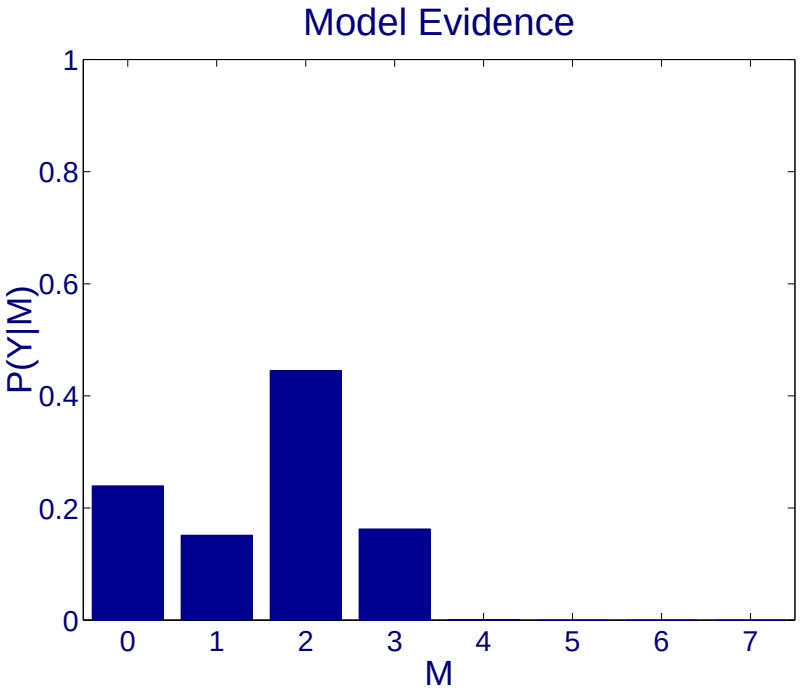
Model classes that are too simple are unlikely to generate the data set.

Model classes that are too complex can generate many possible data sets, so again, they are unlikely to generate that particular data set at random.



Bayesian Model Comparison: Occam's Razor at Work





For example, for quadratic polynomials ($m = 2$): $y = a_0 + a_1 x + a_2 x^2 + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and parameters $\boldsymbol{\theta} = (a_0\ a_1\ a_2\ \sigma)$

`demo: polybayes`

Parametric vs Nonparametric Models

- *Parametric models* assume some finite set of parameters θ. Given the parameters, future predictions, x, are independent of the observed data, $\mathcal{D}$:

$$P(x|\theta, \mathcal{D}) = P(x|\theta)$$

  therefore θ capture everything there is to know about the data.

- So the complexity of the model is bounded even if the amount of data is unbounded. This makes them not very flexible.

- *Non-parametric models* assume that the data distribution cannot be defined in terms of such a finite set of parameters. But they can often be defined by assuming an *infinite dimensional* θ. Usually we think of θ as a *function*.

- The amount of information that θ can capture about the data $\mathcal{D}$ can grow as the amount of data grows. This makes them more flexible.

Bayesian nonparametrics

A simple framework for modelling complex data.

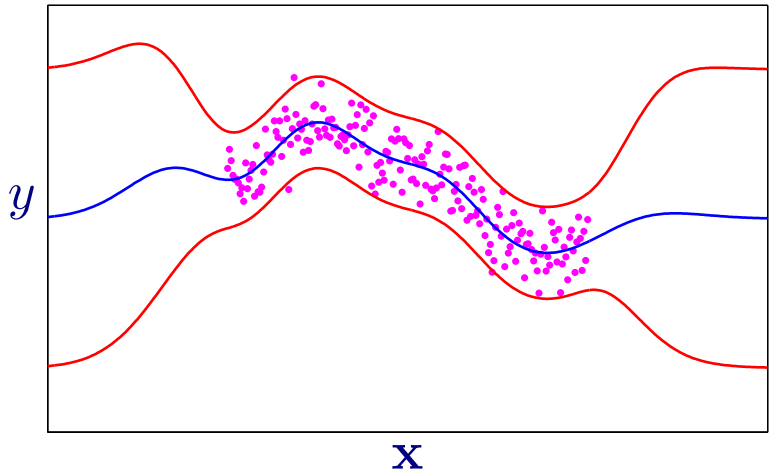
Nonparametric models can be viewed as having infinitely many parameters

Examples of non-parametric models:

Parametric	Non-parametric	Application
polynomial regression	Gaussian processes	function approx.
logistic regression	Gaussian process classifiers	classification
mixture models, k-means	Dirichlet process mixtures	clustering
hidden Markov models	infinite HMMs	time series
factor analysis / pPCA / PMF	infinite latent factor models	feature discovery
...		

Nonlinear regression and Gaussian processes

Consider the problem of nonlinear regression:
You want to learn a function f with error bars from data $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$



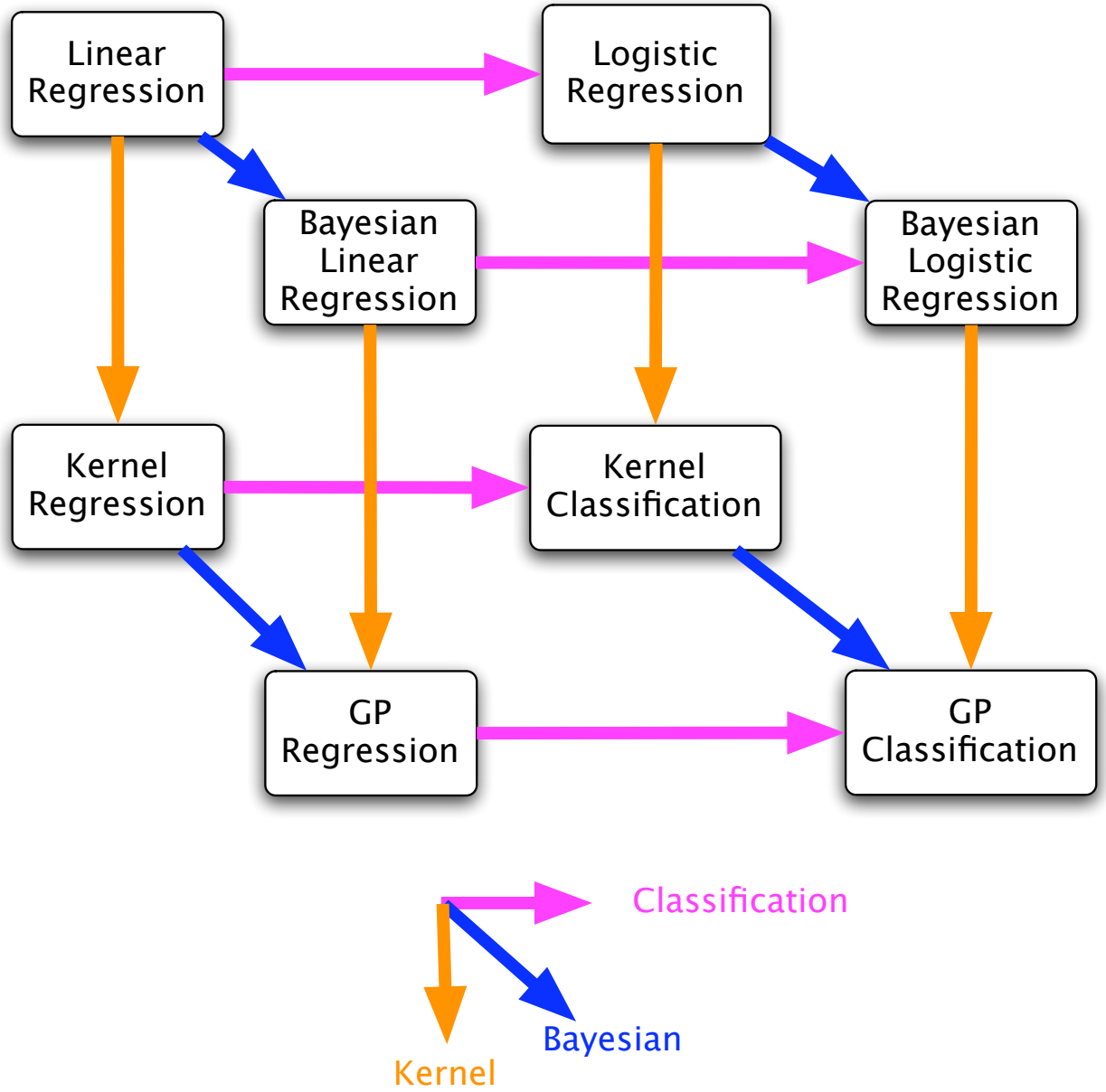
A Gaussian process defines a distribution over functions $p(f)$ which can be used for Bayesian regression:

$$p(f|\mathcal{D}) = \frac{p(f)p(\mathcal{D}|f)}{p(\mathcal{D})}$$

Let $\mathbf{f} = (f(x_1), f(x_2), \ldots, f(x_n))$ be an n-dimensional vector of function values evaluated at n points $x_i \in \mathcal{X}$. Note, $\mathbf{f}$ is a random variable.

Definition: $p(f)$ is a Gaussian process if for *any* finite subset $\{x_1, \ldots, x_n\} \subset \mathcal{X}$, the marginal distribution over that subset $p(\mathbf{f})$ is multivariate Gaussian.

A picture


Linear Regression
Logistic Regression
Bayesian Linear Regression
Bayesian Logistic Regression
Kernel Regression
Kernel Classification
GP Regression
GP Classification
Classification
Bayesian
Kernel

Gaussian process covariance functions (kernels)

$p(f)$ is a Gaussian process if for *any* finite subset $\{x_1, \ldots, x_n\} \subset \mathcal{X}$, the marginal distribution over that finite subset $p(\mathbf{f})$ has a multivariate Gaussian distribution.

Gaussian processes (GPs) are parameterized by a mean function, $\mu(x)$, and a covariance function, or kernel, $K(x, x')$.

$$p(f(x), f(x')) = \mathsf{N}(\boldsymbol{\mu}, \Sigma)$$

where

$$\boldsymbol{\mu} = \begin{bmatrix} \mu(x) \\ \mu(x') \end{bmatrix} \quad \Sigma = \begin{bmatrix} K(x,x) & K(x,x') \\ K(x',x) & K(x',x') \end{bmatrix}$$

and similarly for $p(f(x_1), \ldots, f(x_n))$ where now $\boldsymbol{\mu}$ is an $n \times 1$ vector and Σ is an $n \times n$ matrix.

Gaussian process covariance functions

Gaussian processes (GPs) are parameterized by a mean function, $\mu(x)$, and a covariance function, $K(x, x')$, where $\mu(x) = \mathsf{E}(f(x))$ and $K(x, x') = \mathsf{Cov}(f(x), f(x'))$.

An example covariance function:

$$K(x, x') = v_0 \exp\left\{-\left(\frac{|x - x'|}{r}\right)^{\alpha}\right\} + v_1 + v_2\,\delta_{ij}$$

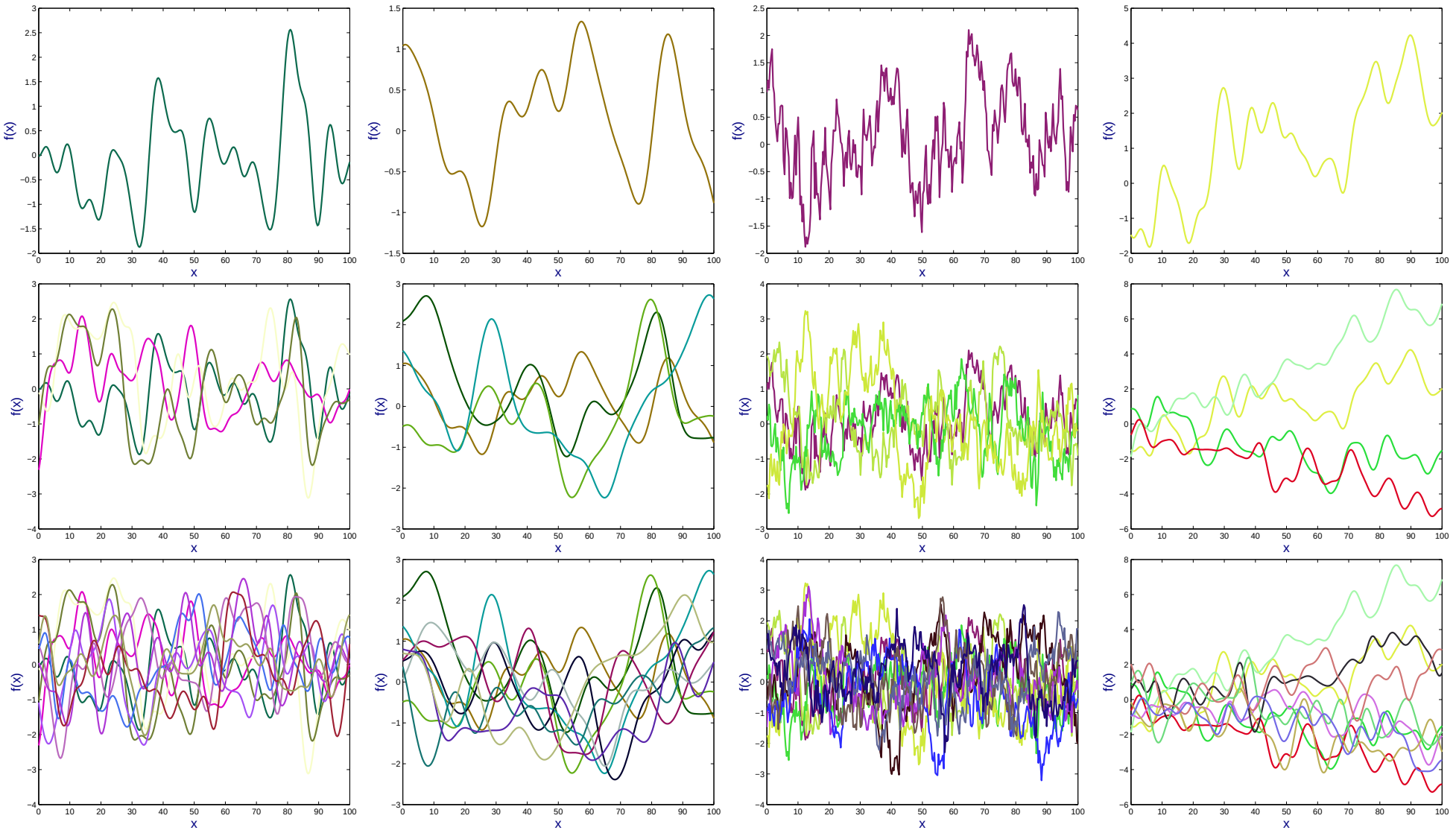
with parameters $(v_0, v_1, v_2, r, \alpha)$.

These kernel parameters are interpretable and can be learned from data:

v_0	signal variance
v_1	variance of bias
v_2	noise variance
r	lengthscale
α	roughness

Once the mean and covariance functions are defined, everything else about GPs follows from the basic rules of probability applied to mutivariate Gaussians.

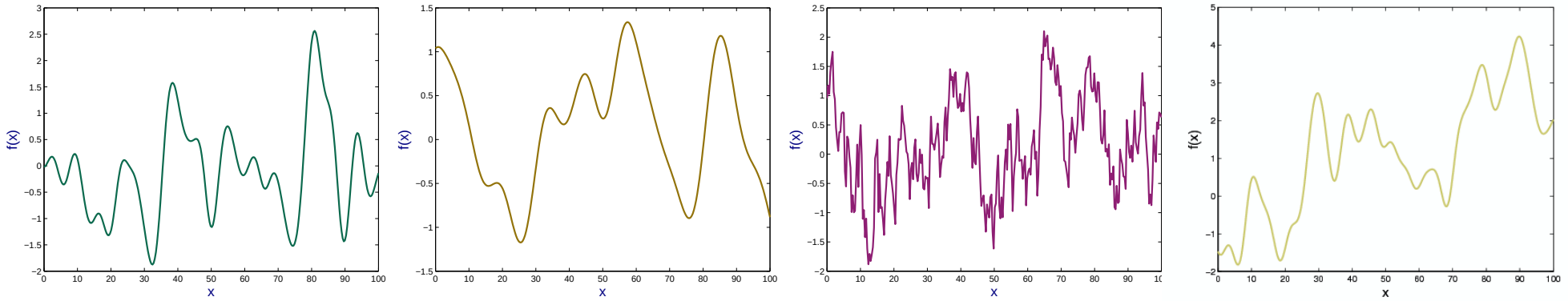
Samples from GPs with different $K(x, x')$



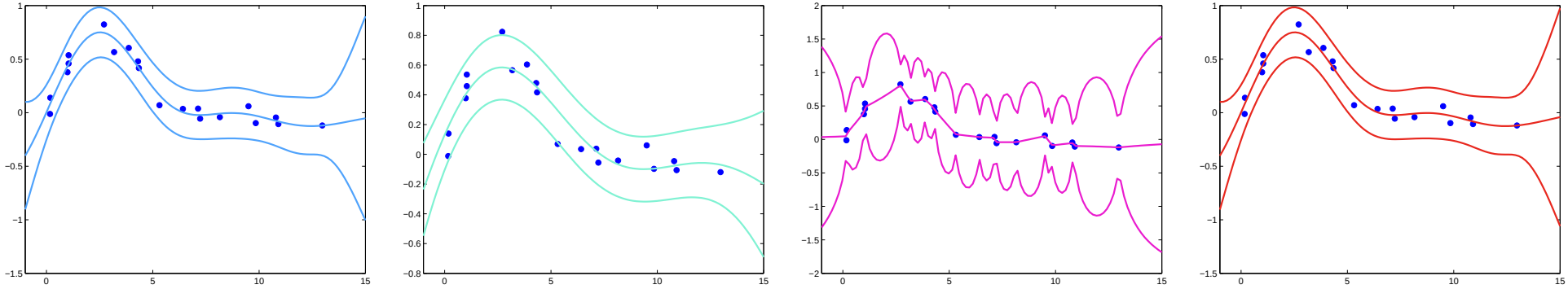
gpdemogen

Prediction using GPs with different $K(x, x')$

A sample from the prior for each covariance function:



Corresponding predictions, mean with two standard deviations:



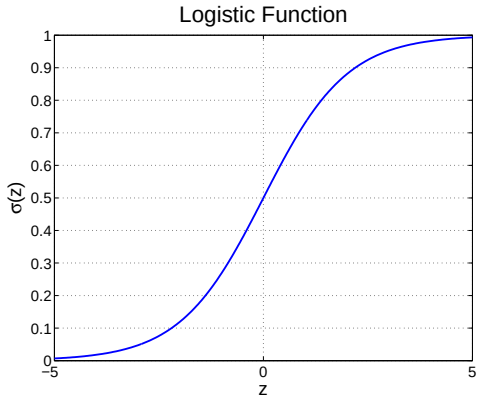
gpdemo

Change Point Kernels

Assume $f_1(x) \sim GP(0, k_1)$ and $f_2(x) \sim GP(0, k_2)$. Define:

$$f(x) = (1 - \sigma(x))f_1(x) + \sigma(x)f_2(x)$$

where σ is a sigmoid function between 0 and 1, such as the logistic function: $\sigma(x) = 1/(1 + \exp(-x))$



Then $f \sim GP(0, k)$, where

$$k(x, x') = (1 - \sigma(x))\, k_1(x, x')\, (1 - \sigma(x')) + \sigma(x)\, k_2(x, x')\, \sigma(x')$$

We can parametrise the location τ and abruptness a of the changepoint by replacing $\sigma(x)$ with $\sigma(a(x - \tau))$.

Intutively (in one-dimension), the function f behaves like f_1 before τ and like f_2 after τ.

(cf. Garnett, Osborne and Roberts, 2009)

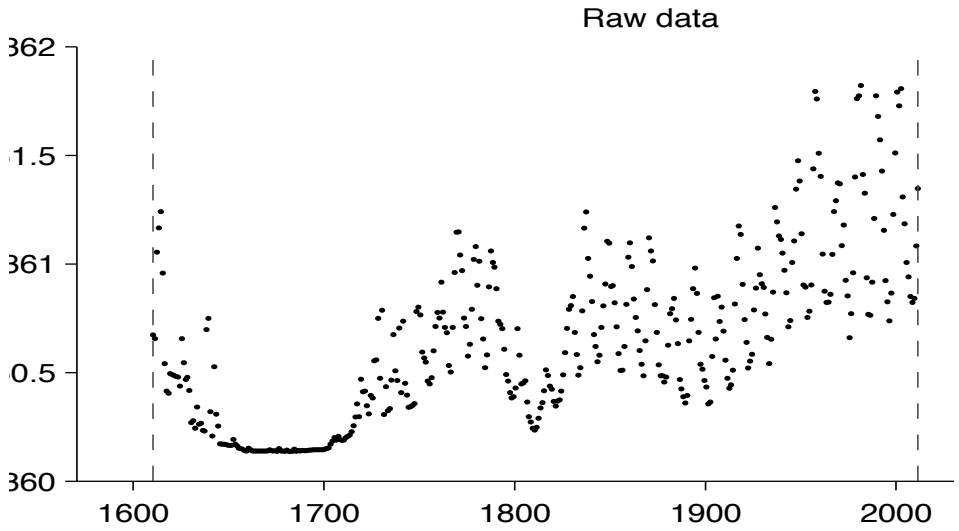
Change Windows

A **change window** (or interval) is simply defined as a changepoint from $f_1(x)$ to $f_2(x)$ followed by a changepoint back to $f_1(x)$.

We can represent this via a product of two sigmoids with different offsets:

$$f(x) = (1-(1-\sigma(a(x-\tau_2)))\sigma(a(x-\tau_1)))f_1(x)+(1-\sigma(a(x-\tau_2)))\sigma(a(x-\tau_1))f_2(x)$$

This looks a bit messy but it just smoothly switches on f_2 between τ_1 and τ_2.



Solar irradiance data form 1600s showing the Maunder minimum where sunspot activity was extremely rare.

The Automatic Statistician

How do we learn the kernel?

- **Usual approach:** parametrise the kernel with a few hyperparameters and optimise or infer these. An example covariance function:

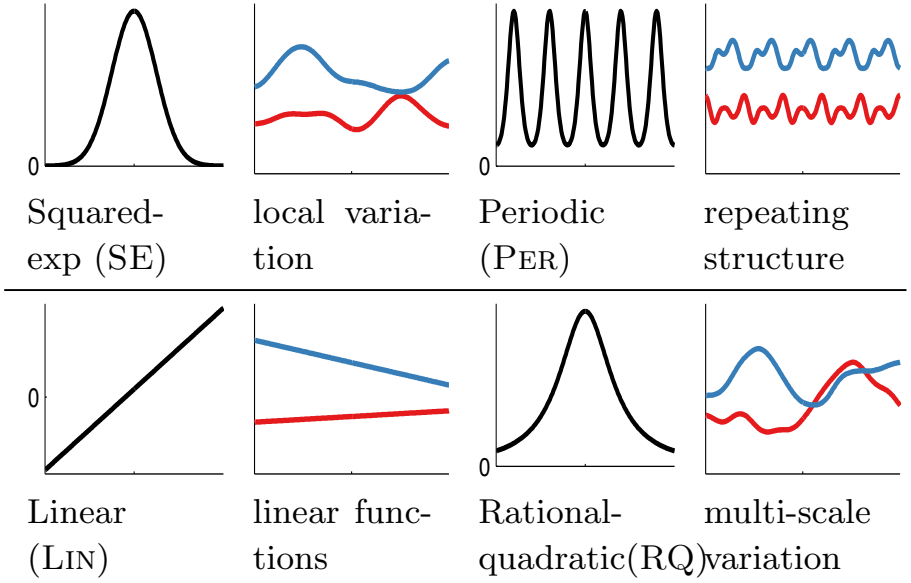
$$K(x, x') = v_0 \exp\left\{-\left(\frac{|x - x'|}{r}\right)^{\alpha}\right\} + v_1 + v_2\,\delta_{ij}$$

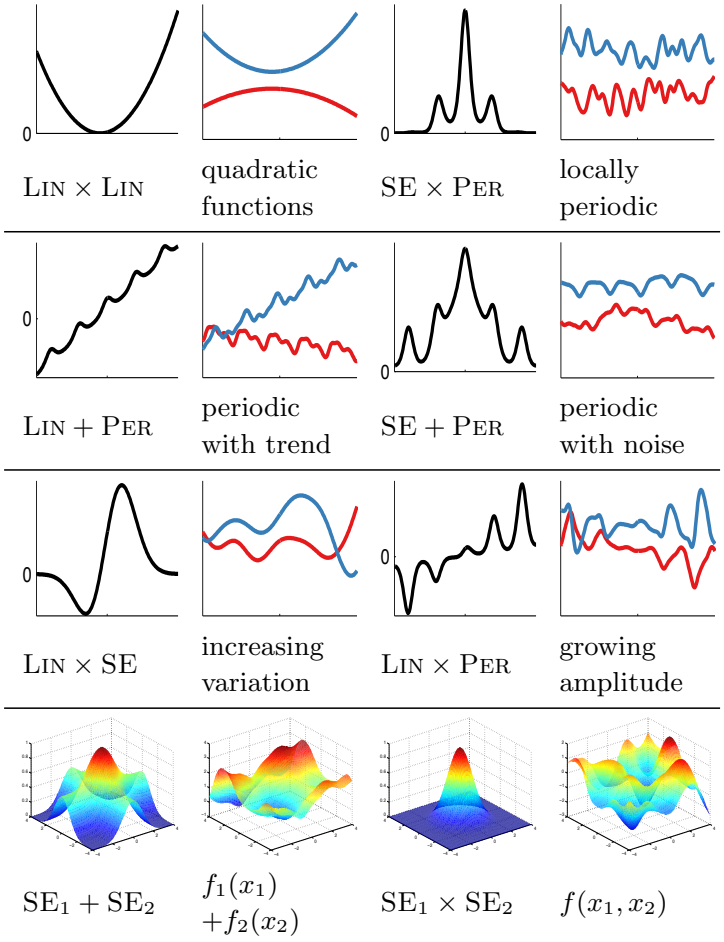
  with parameters $(v_0, v_1, v_2, r, \alpha)$.

- **Our approach:** Define a grammar over kernels and search over this grammar to discover an appropriate and interpretable structure of the kernel.

Kernel Composition

By taking a few simple **base kernels** and two **composition rules**, *kernel addition and multiplication*, we can span a rich space of structured kernels.





(w/ Duvenaud, Lloyd, Grosse, and Tenenbaum, ICML 2013)

see also (Wilson and Adams, ICML 2013)

Kernel Composition

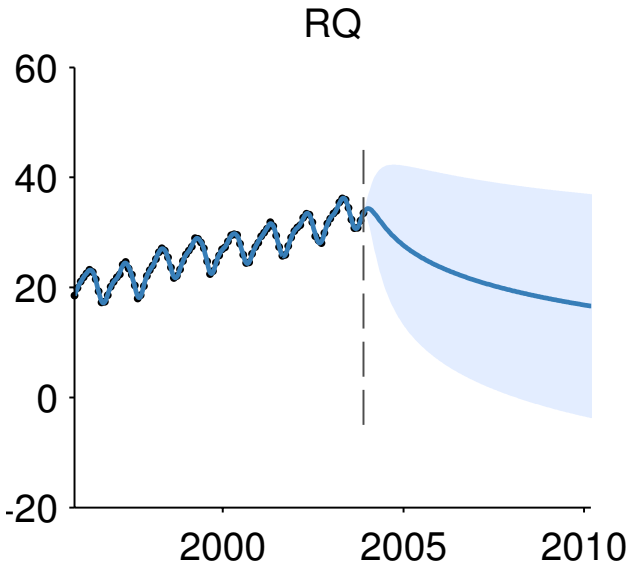
Kernel	Function description
WN	White noise
C	Constant
LIN	Linear
SE	Smooth
PER	Periodic

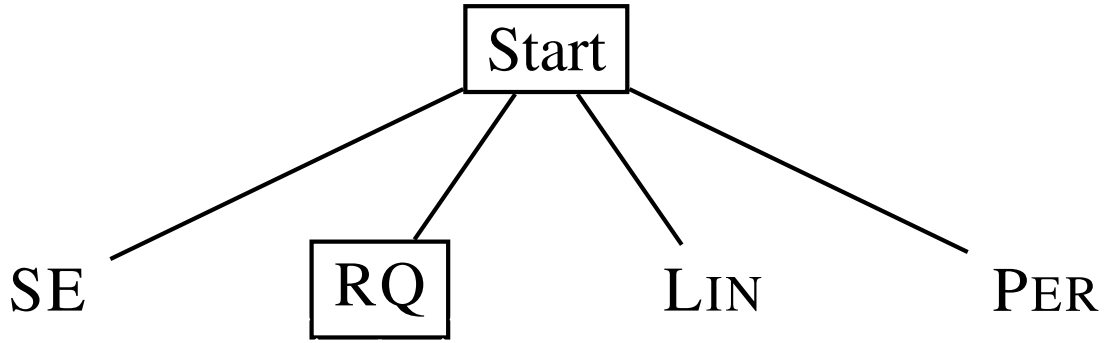
Motif	Example syntax
Linear regression	$\mathrm{C} + \mathrm{LIN}$
Fourier analysis	$\mathrm{C} + \sum \cos$
Sparse spectrum GPs	$\sum \cos$
Spectral kernels	$\sum \mathrm{SE} \times \cos$
Changepoints	e.g. $\mathrm{CP}(\mathrm{SE}, \mathrm{SE})$
Kernel smoothing	SE
Heteroscedasticity	e.g. $\mathrm{SE} + \mathrm{LIN} \times \mathrm{WN}$
Trend cyclical irregular	$\sum \mathrm{SE} + \sum \mathrm{PER}$
Additive nonparametric	$\sum \mathrm{SE}$

Different kernels express a variety of covariance structures, such as local smoothness or periodicity. New kernels can be constructed by taking the product of a set of base kernels to express richer structures, (e.g. locally periodic, or heteroscedastic)

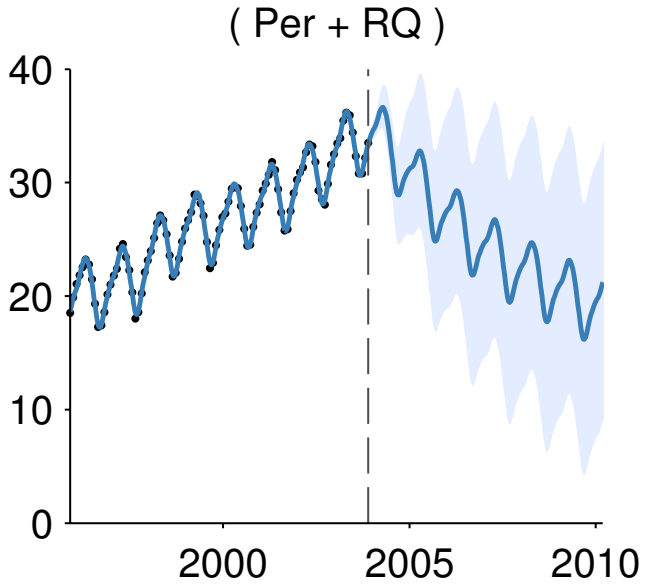
- Search starts with the **base kernels** for the GP, and applies different operations (addition, multiplication, CP, CW) to explore kernels spanned by the grammar.
- For efficiency, kernel hyperparameters are optimised rather than integrated out
- Each resulting model is scored using the **marginal likelihood** penalised by a BIC term for number of hyperparameters.

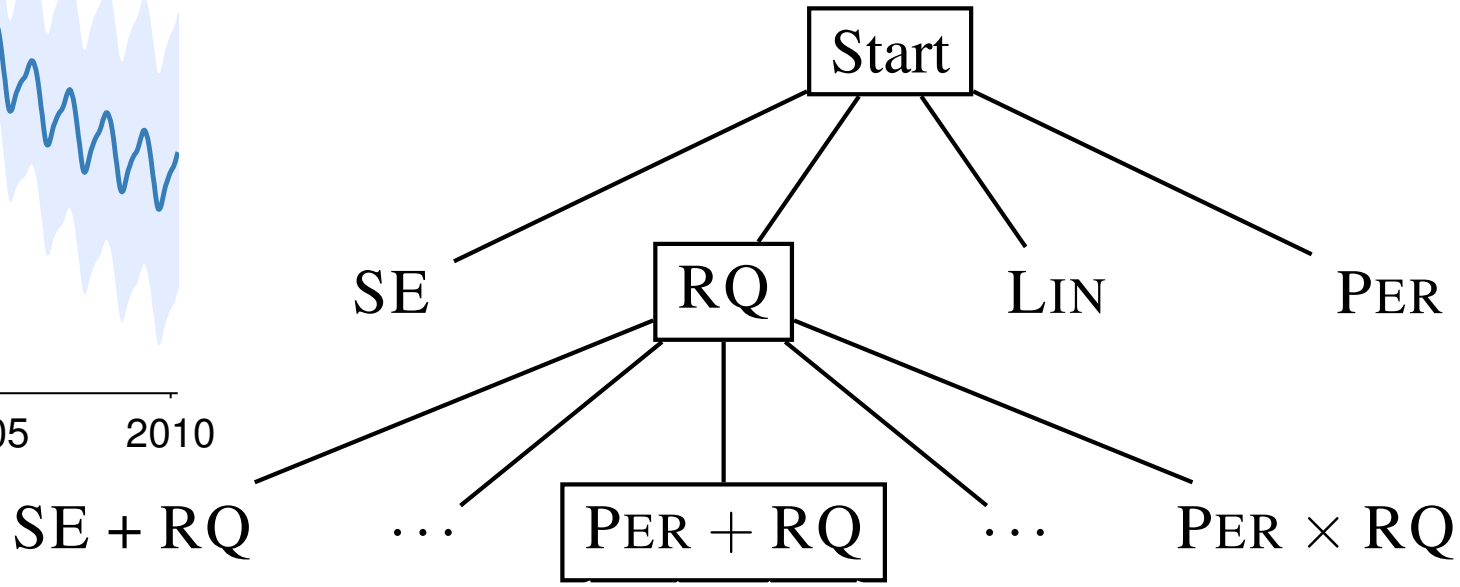
Kernel Composition: Mauna Loa CO_2 Keeling Curve



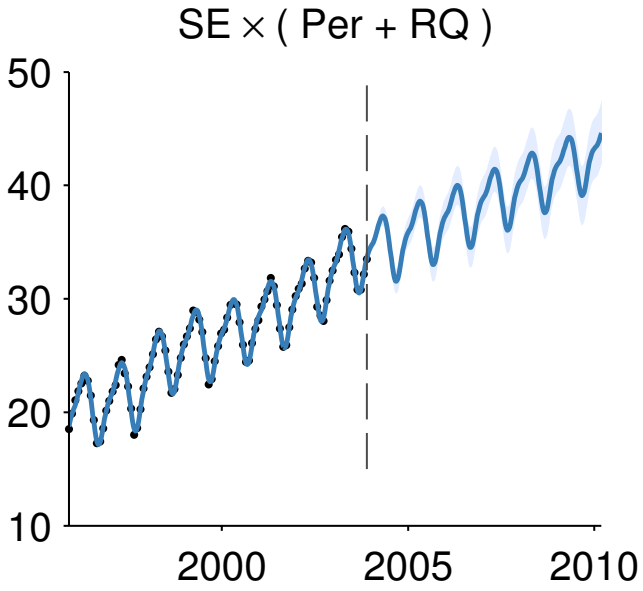


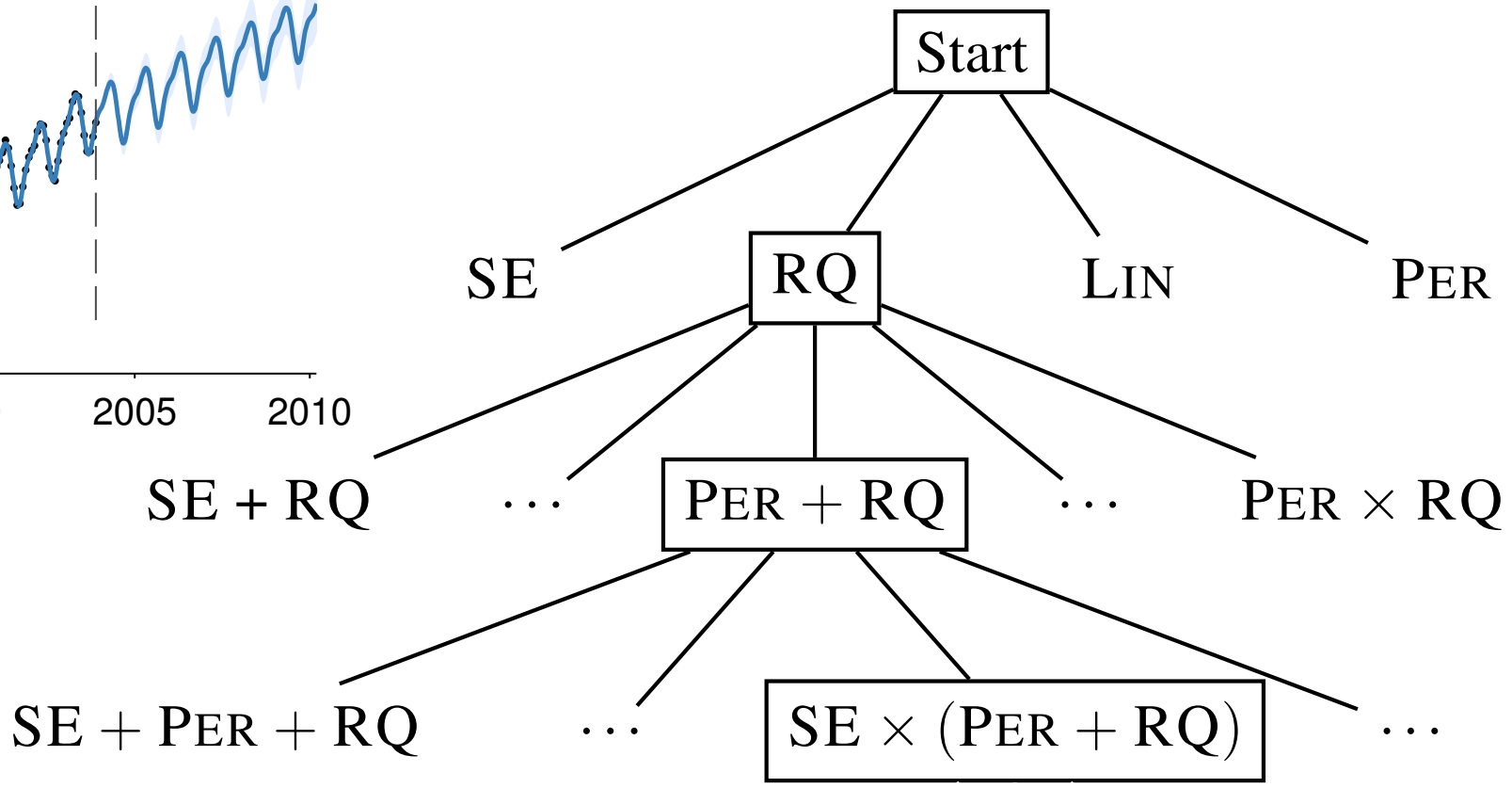
Kernel Composition: Mauna Loa CO_2 Keeling Curve



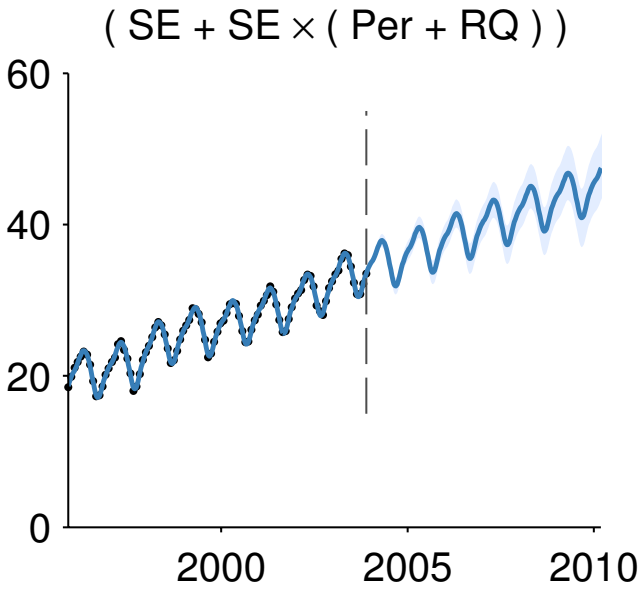


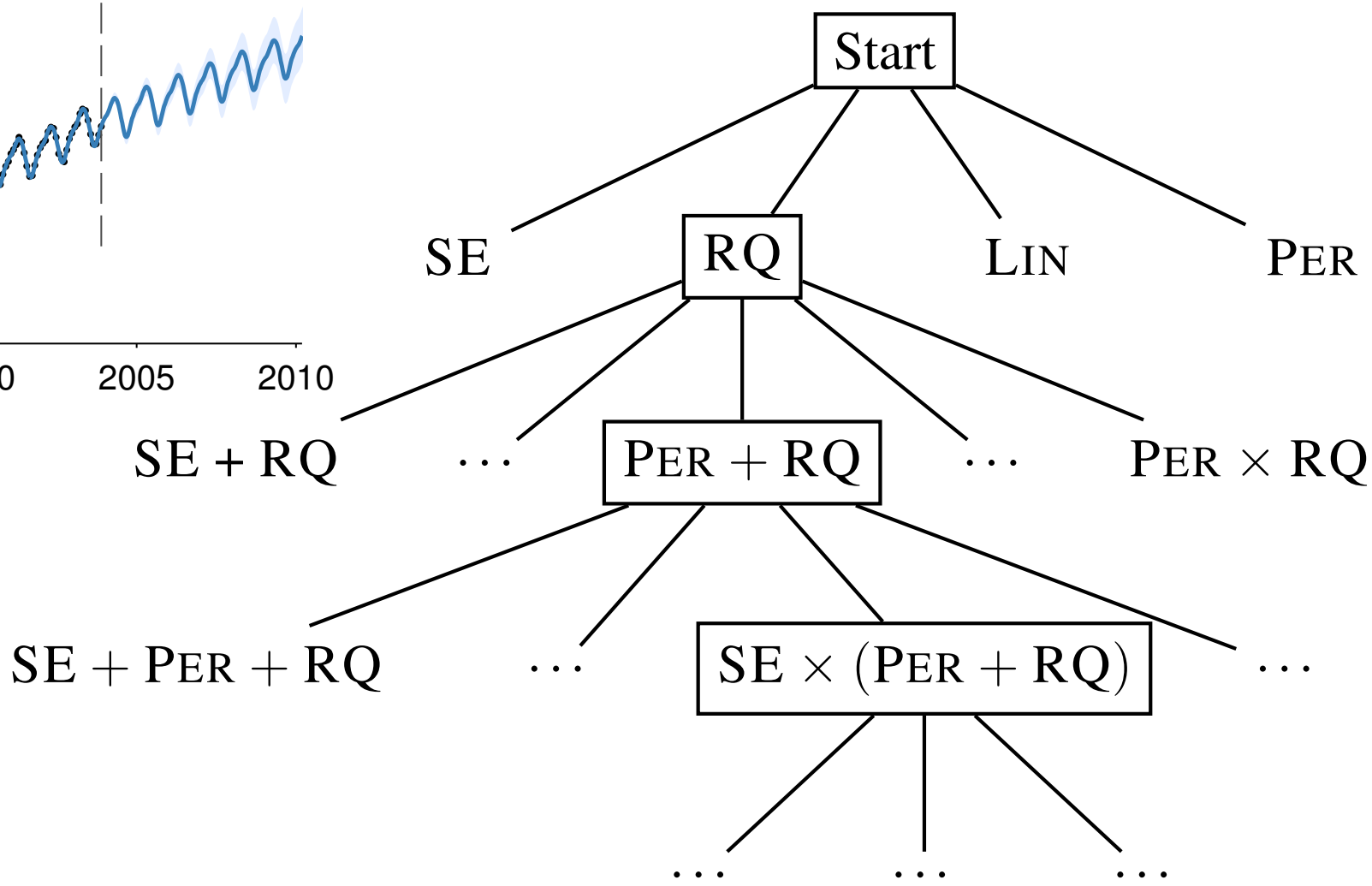
Kernel Composition: Mauna Loa CO_2 Keeling Curve



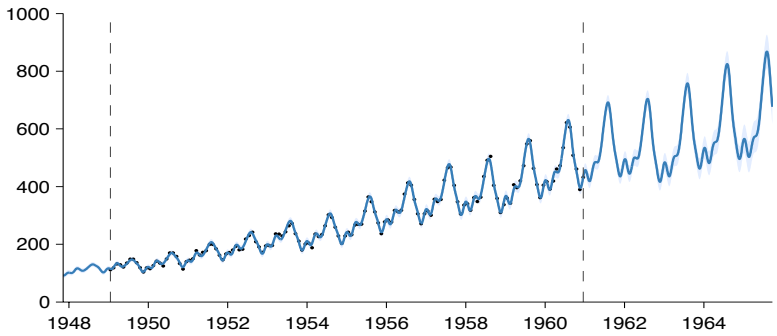


Kernel Composition: Mauna Loa CO_2 Keeling Curve

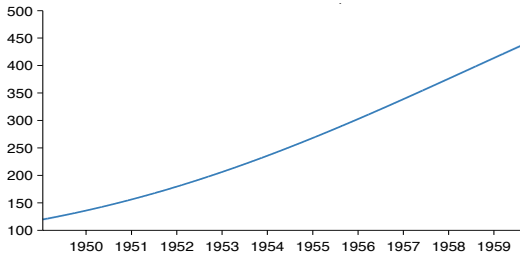




Example: An automatic analysis

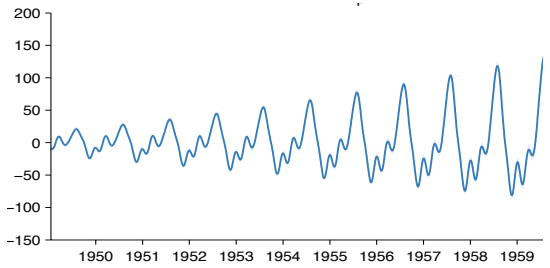


=



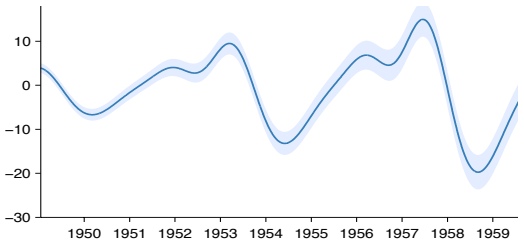
A very smooth, monotonically increasing function

+



An approximately periodic function with a period of 1.0 years and with approximately linearly increasing amplitude

+

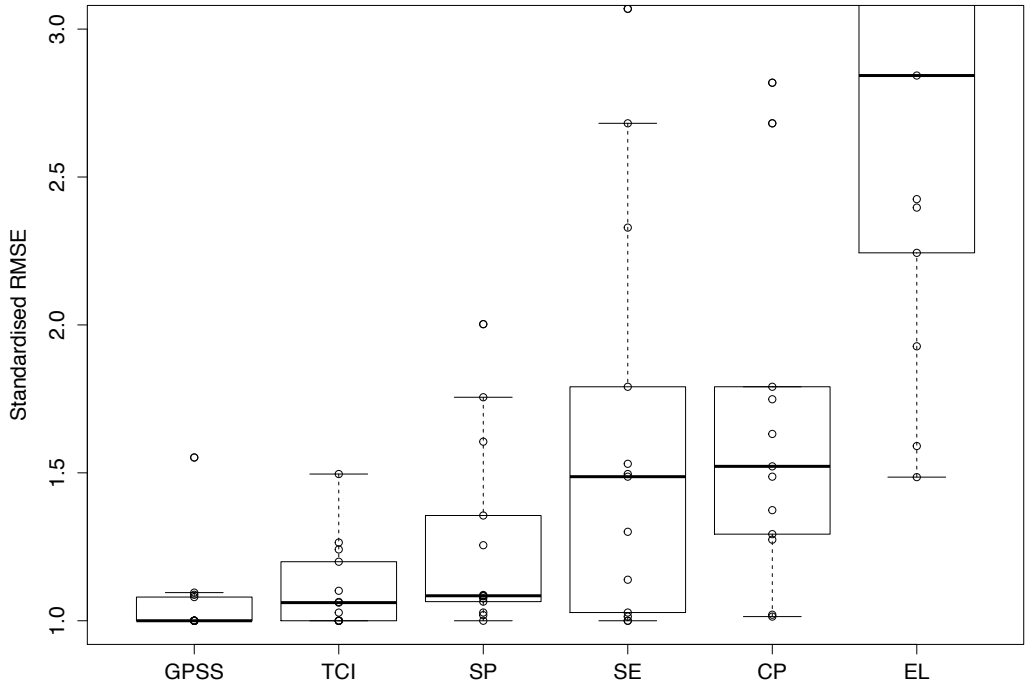


An exactly periodic function with a period of 4.3 years but with linearly increasing amplitude

Kernel Composition: predictive results

	Mean Squared Error (MSE)					Negative Log-Likelihood				
Method	bach	concrete	puma	servo	housing	bach	concrete	puma	servo	housing
Linear Regression	1.031	0.404	0.641	0.523	0.289	2.430	1.403	1.881	1.678	1.052
GAM	1.259	0.149	0.598	0.281	0.161	1.708	0.467	1.195	0.800	0.457
HKL	**0.199**	0.147	0.346	0.199	0.151	-	-	-	-	-
GP SE-ARD	**0.045**	0.157	0.317	0.126	**0.092**	**−0.131**	0.398	0.843	0.429	0.207
GP Additive	**0.045**	**0.089**	**0.316**	**0.110**	0.102	**−0.131**	**0.114**	**0.841**	**0.309**	0.194
Structure Search	**0.044**	**0.087**	**0.315**	**0.102**	**0.082**	**−0.141**	**0.065**	**0.840**	**0.265**	**0.059**

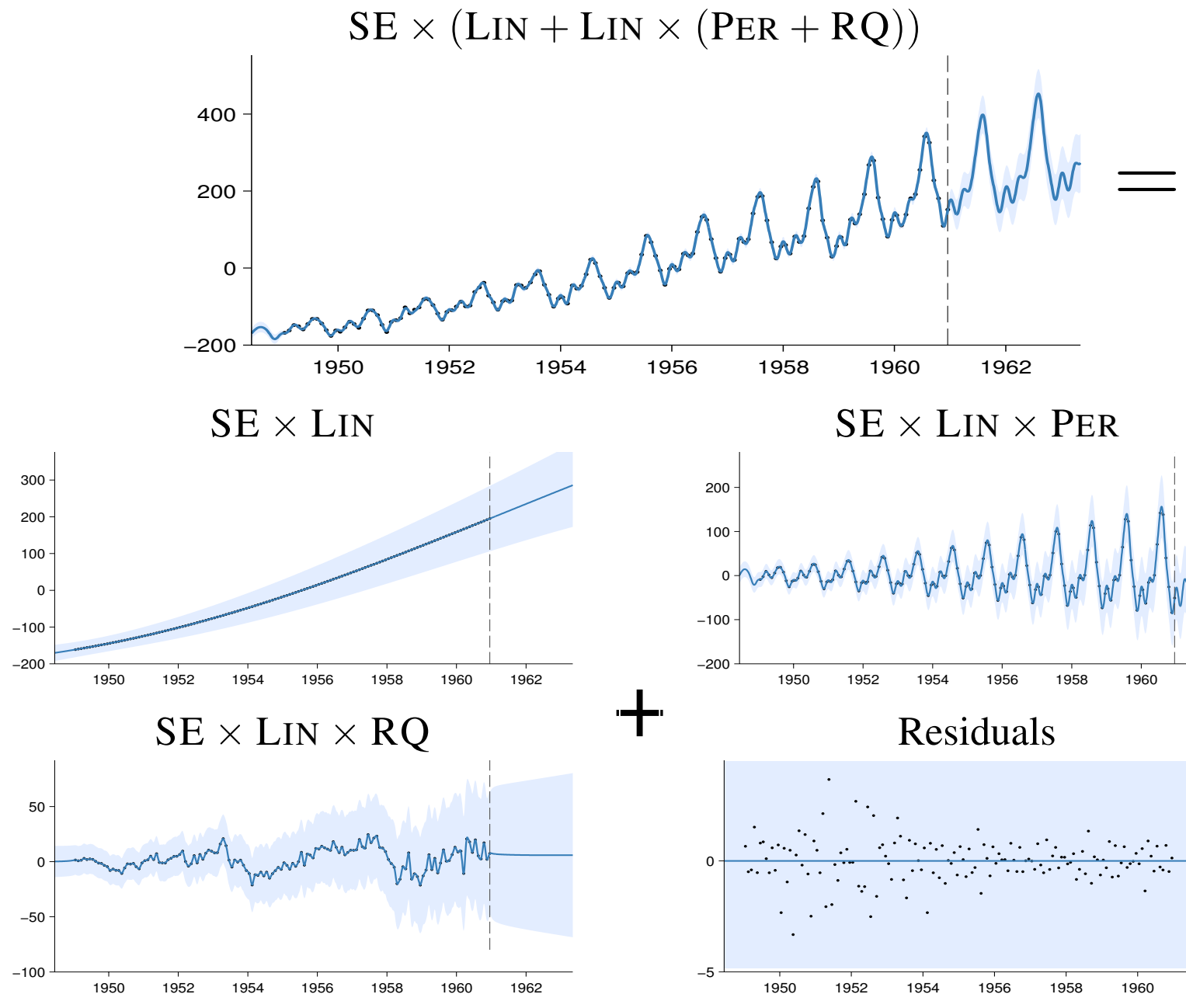
Predictive Results



Combined results over 13 time series data sets comparing 5 methods

GPSS:	Automatic Statistician using Gaussian process structure search
TCI:	trend-cyclical-irregular models (e.g. Lind et al., 2006)
SP:	spectral kernels (Wilson & Adams, 2013)
SE:	additive nonparametric regression (SE) (e.g. Buja et al., 1989)
CP:	change point modelling (CP) / multi resolution GP (e.g. Garnett et al., 2009)
EL:	equation learning using Eureqa (Nutonian, 2011)

Distributivity helps interpretability



Generating text output from the Automatic Statistician

The structure search algorithm has identified nine additive components in the data. The first 4 additive components explain 92.3% of the variation in the data as shown by the coefficient of determination (R^2) values in table 1. The first 8 additive components explain 99.2% of the variation in the data. After the first 5 components the cross validated mean absolute error (MAE) does not decrease by more than 0.1%. This suggests that subsequent terms are modelling very short term trends, uncorrelated noise or are artefacts of the model or search procedure. Short summaries of the additive components are as follows:

- A constant.
- A constant. This function applies from 1644 until 1713.
- A smooth function. This function applies until 1644 and from 1719 onwards.
- An approximately periodic function with a period of 10.8 years. This function applies until 1644 and from 1719 onwards.
- A rapidly varying smooth function. This function applies until 1644 and from 1719 on-

This component is constant. This component applies from 1644 until 1713.

This component explains 35.3% of the residual variance; this increases the total variance explained from 0.0% to 35.3%. The addition of this component reduces the cross validated MAE by 29.42% from 0.33 to 0.23.

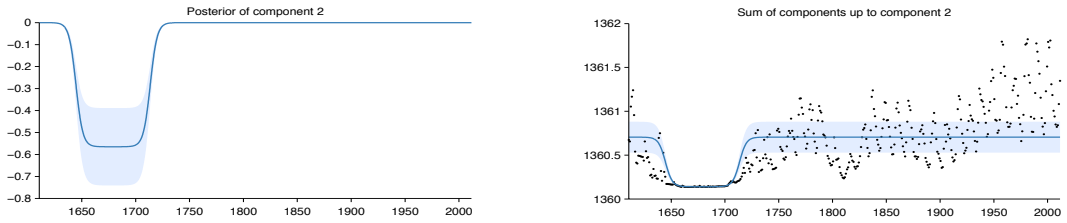


Figure 3: Posterior of component 2 (left) and the posterior of the cumulative sum of components with data (right)

"Automatic Construction and Natural-language Description of Additive Nonparametric Models"

(w/ James Lloyd, David Duvenaud, Roger Grosse and Josh Tenenbaum, NIPS workshop, 2013)

Example reports

01-airline.pdf
02-solar.pdf
07-call-centre.pdf
09-gas-production.pdf

Challenges

- Trading off predictive performance and interpretability
- Expressing a large and flexible enough class of models so that different kinds of data can be captured
- The computational complexity of searching a huge space of models
- Translating complex modelling constructs into the English language; automatically generating relevant visualisations

Current and Future Directions

- Automatic Statistician for:
  - multivariate nonlinear regression $y = f(\mathbf{x})$
  - classification
  - completing and interpreting tables and databases

- Probabilistic Programming
  - probabilistic models are expressed in a general (Turing complete) programming language (e.g. Church/Venture/Anglican)
  - a **universal inference engine** can then be used to infer unobserved variables given observed data
  - this can be used to implement seach over the model space in an automated statistician

Summary

- The Automatic Statistician project aims to automate certain kinds of exploratory and predictive modelling
- Conceptually, we follow a Bayesian framework, relying in particular on Bayesian nonparametric models for flexibility
- The ultimate aim is to produce output that is interpretable by a reasonably numerate non-statistician
- We have a system that can produce readable 10-15 page reports from one dimensional time series, capturing non-stationarity, change-points and change-windows, periodicity, trends, etc
- Predictive performance seems very competitive

Thanks


James Lloyd


David Duvenaud


Roger Grosse

Duvenaud, D., Lloyd, J. R., Grosse, R., Tenenbaum, J. B. and Ghahramani, Z. (2013) Structure Discovery in Nonparametric Regression through Compositional Kernel Search. ICML 2013.

Lloyd, J. R., Duvenaud, D., Grosse, R., Tenenbaum, J. B. and Ghahramani, Z. (2013) Automatic Construction and Natural-language Description of Additive Nonparametric Models. NIPS workshop on Constructive Machine Learning

Ghahramani, Z. (2013) Bayesian nonparametrics and the probabilistic approach to modelling. *Philosophical Trans. Royal Society A* 371: 20110553.

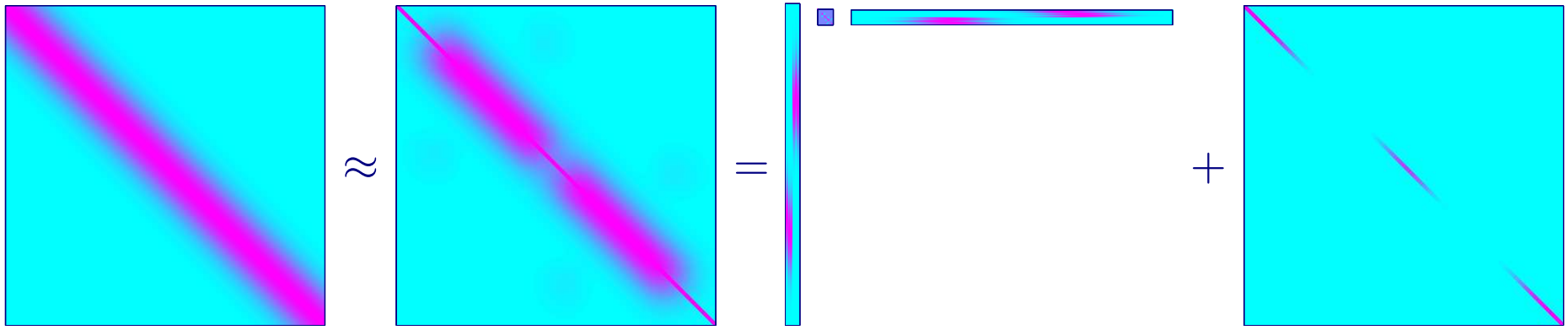
Appendix

Speeding up GP learning: Inducing point approximations

(Snelson and Ghahramani, 2006)

We can approximate GP through $M < N$ **inducing points** $\bar{\mathbf{f}}$ to obtain the Sparse Pseudo-input Gaussian process (SPGP) a.k.a. FITC: $p(\mathbf{f}) = \int \mathrm{d}\bar{\mathbf{f}} \prod_n p(f_n|\bar{\mathbf{f}})\, p(\bar{\mathbf{f}})$

GP prior $\mathcal{N}(\mathbf{0}, \mathbf{K}_N)$ $\approx$ $p(\mathbf{f})$ $= \mathcal{N}(\mathbf{0}, \mathbf{K}_{NM}\mathbf{K}_M^{-1}\mathbf{K}_{MN} + \mathbf{\Lambda})$ FITC prior



- FITC covariance inverted in $\mathcal{O}(M^2N) \ll \mathcal{O}(N^3) \Rightarrow$ much faster
- FITC $=$ GP with non-stationary covariance parameterized by $\bar{\mathbf{X}}$
- Given data $\{\mathbf{X}, \mathbf{y}\}$ with noise σ^2, predictive mean and variance can be computed in $\mathcal{O}(M)$ and $\mathcal{O}(M^2)$ per test case respectively

Builds on a large lit on sparse GPs (see Quiñonero Candela and Rasmussen, 2006).

Speeding up GP learning: some developments since 2006

- FITC (2006)
- Unifying review (see Quiñonero Candela and Rasmussen, 2006)
- Combining local and global approximations (w/ Snelson, 2007)
- Generalised FITC (for classification) (Naish-Guzman and Holden, 2007)
- Variational learning of inducing variables in sparse GPs (Titsias, 2009)
- Exploiting additive and Kronecker structure of kernels for very fast inference (Saatci, 2011; Gilboa, Saatci, Cunningham 2013)
- GPatt: Fast Multidimensional Pattern Extrapolation with GPs (Wilson, Gilboa, Nehorai, Cunningham, 2013) learns very flexible stationary kernels on 380k points using Kronecker structure
- Gaussian Processes for Big Data (Hensman, Fusi, Lawrence, 2013) uses SVI to handle million data points