Lecture 9

Time series prediction

Prediction is about function fitting

To predict we need to **model**

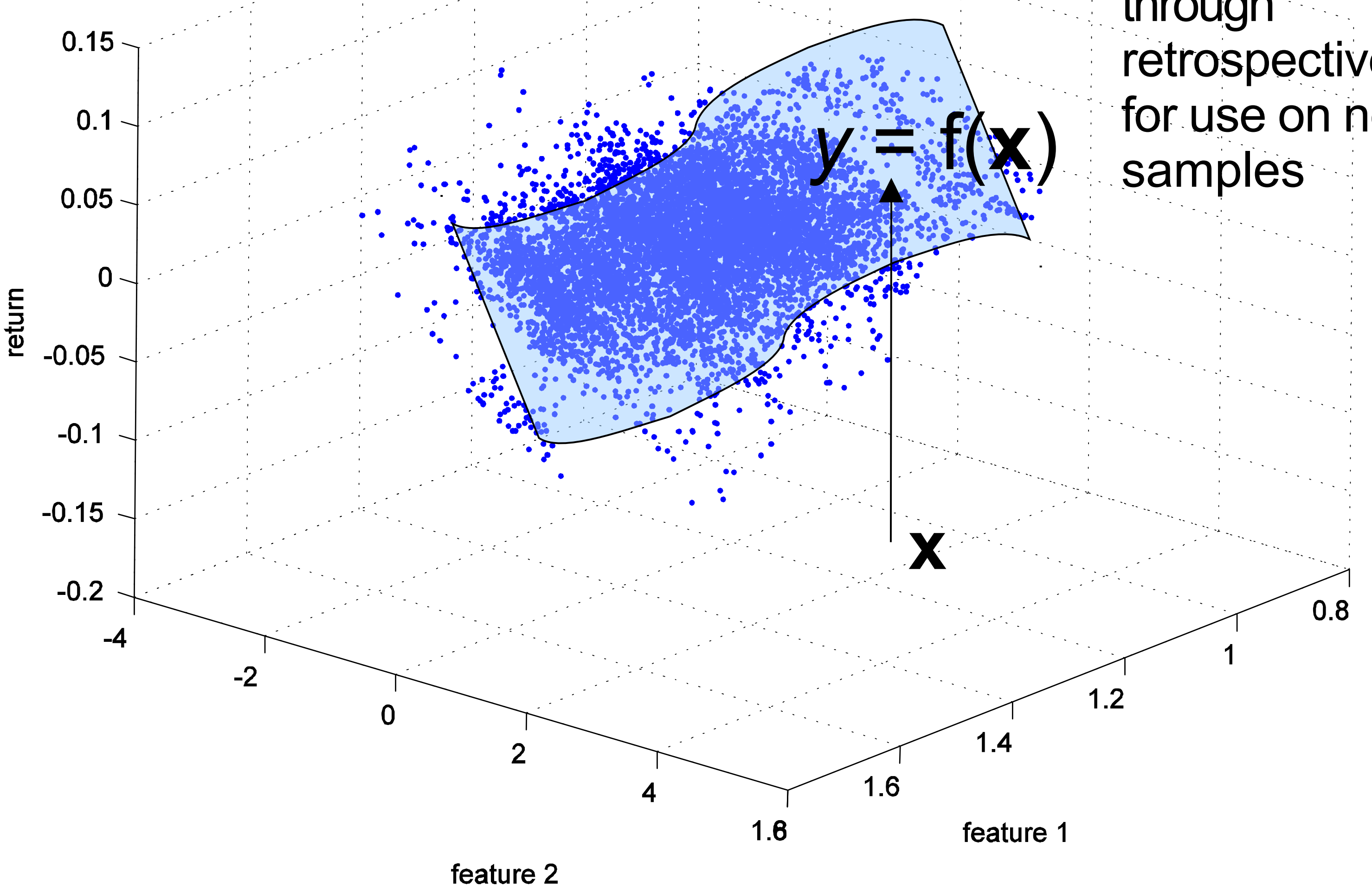
There are a bewildering number of models for data – we look at some of the major approaches in this lecture – but this is far from complete

We'll start by looking at the difference between **function** and **curve** modelling approaches

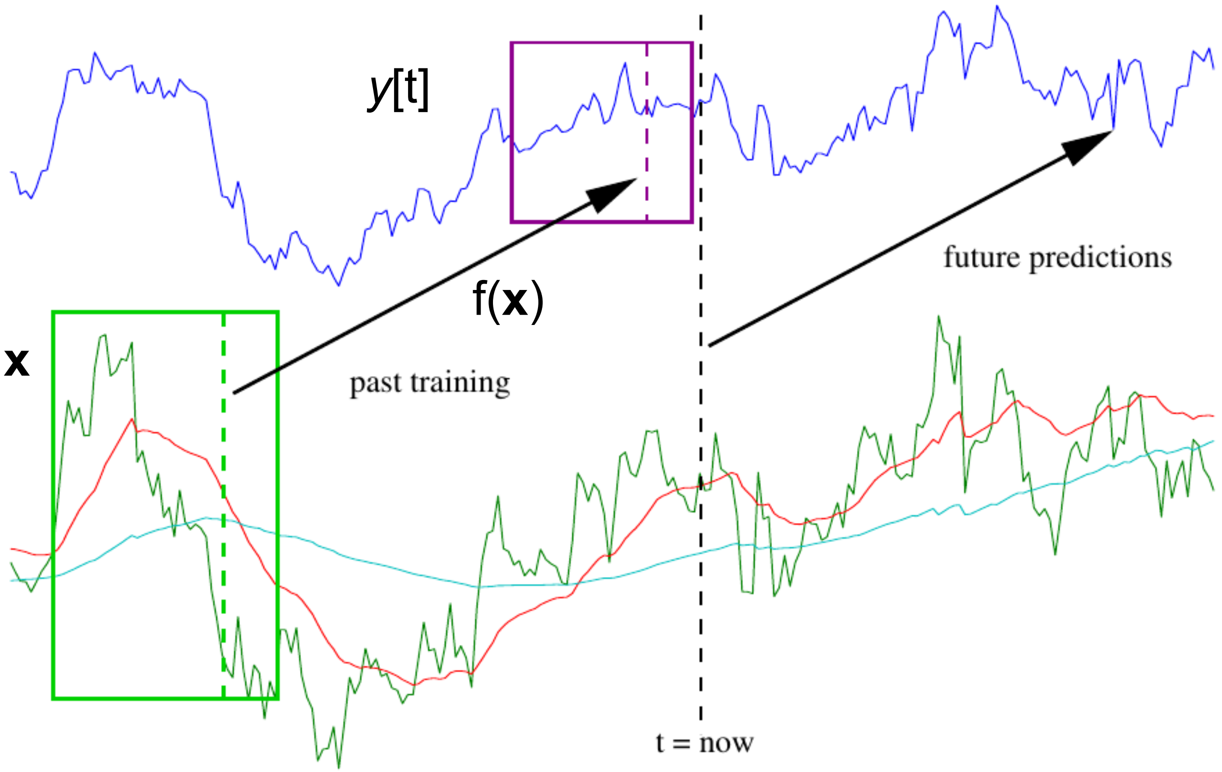
Function mapping

A forecast, *y[t]*, is estimated from a function of observables **x**. The latter are **not necessarily** even from the past of *y[t]*. Critically, **x** is not just the time variable

Infer a manifold or response curve through retrospective data for use on new samples



Example – rolling regression



Training data – develop mapping

Future data – use mapping

x corresponds to a set of past samples, say **x** = (*y*[t-L],...*y*[t-L+h]). We develop a **mapping** from **x** to *y*[t] over a **training set** then use this mapping for subsequent forecasting

The samples in **x** don't even need to come from *y* though – so this is really flexible

What form could the mapping take?

Our mapping function can be **any universal approximation approach** – this will include (but not limited by), for example:

Gaussian processes
Neural networks
Basis function models

and many more...

For example – we already have looked at basis function models $y = \mathbf{w}^{\mathsf{T}}\boldsymbol{\Phi}$

The simple **linear** model has $\boldsymbol{\Phi} = [1, \mathbf{X}]^{\mathsf{T}}$: if we chose those **X** to be the recent *past samples of y,* then this is just the **autoregressive model**

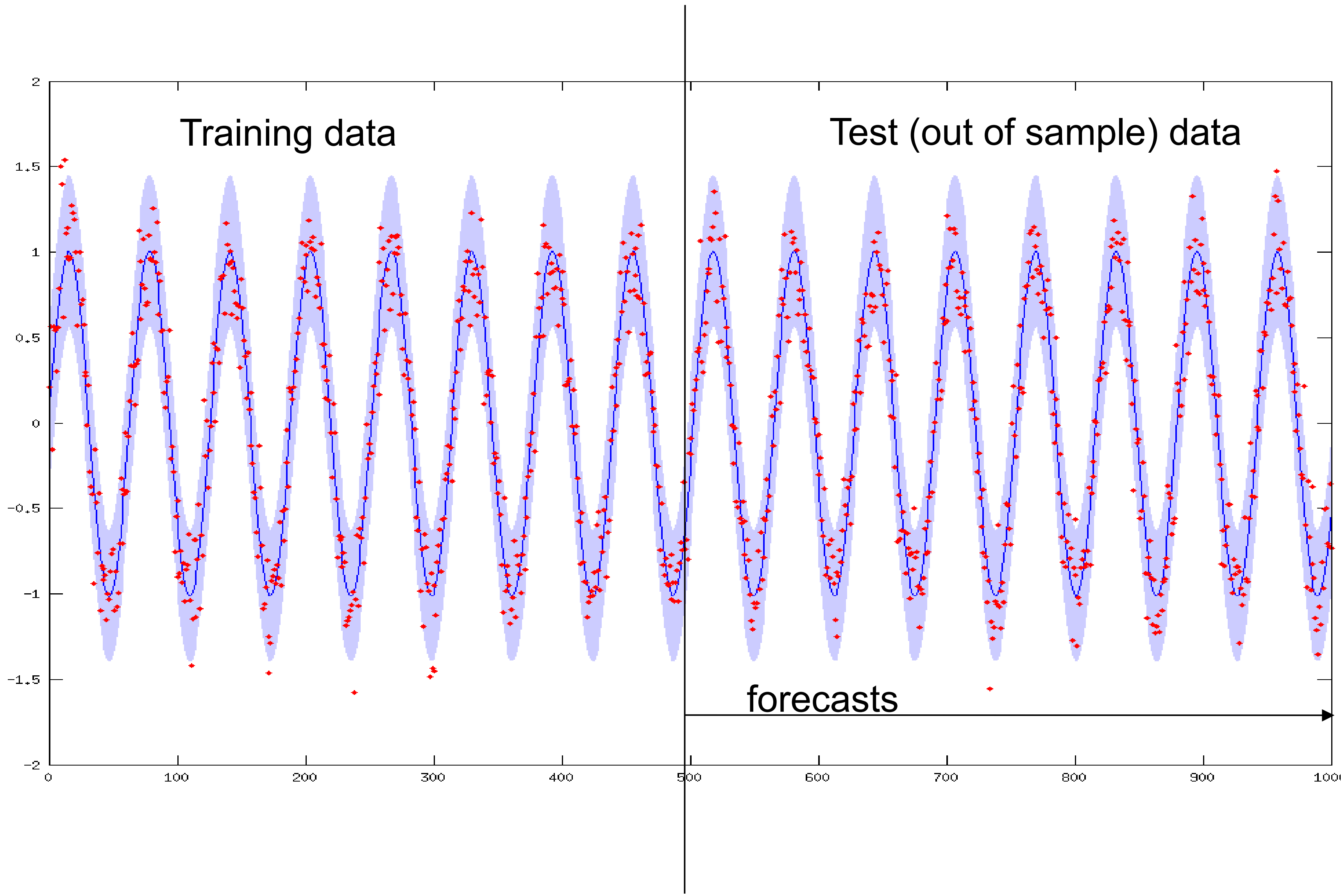
We can trivially extend this to a **non-linear basis**

$$\boldsymbol{\Phi} = [1, \mathbf{X}, \phi_{\text{harmonics}}(\mathbf{X})]^{\mathsf{T}}$$

$$\boldsymbol{\Phi} = [1, \mathbf{X}, \phi_{\text{Gaussians}}(\mathbf{X}), \phi_{\text{harmonics}}(\mathbf{X})]^{\mathsf{T}}$$

Simple example

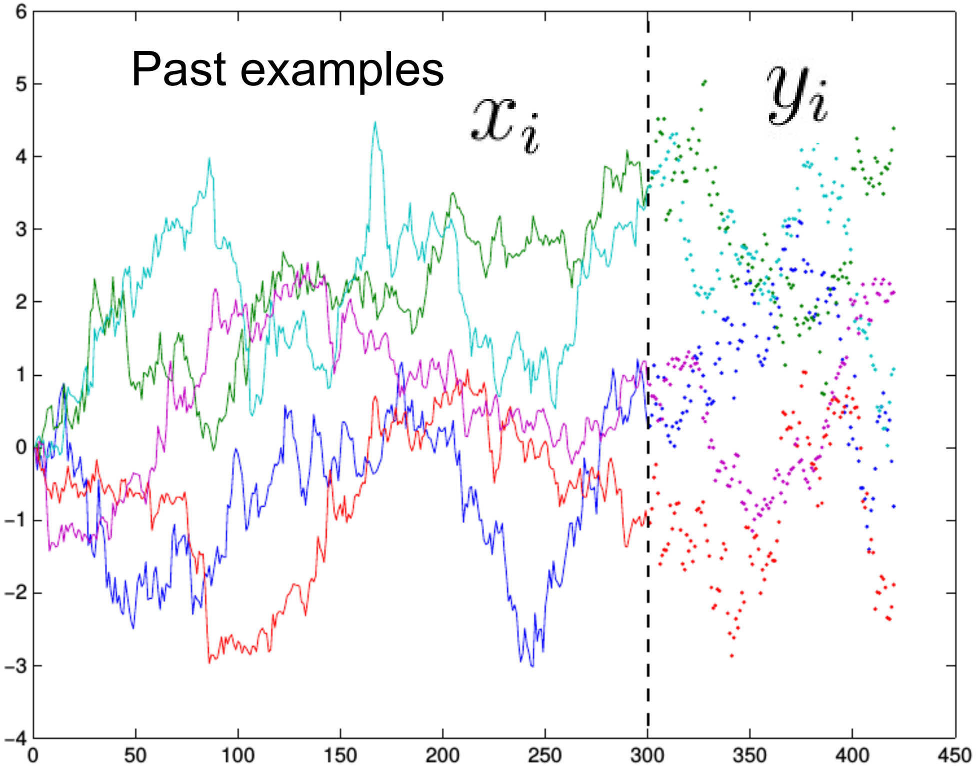
$$y = \mathbf{w}^{\mathsf{T}}\mathbf{\Phi} \qquad \mathbf{\Phi} = [1, \mathbf{X}, \phi_{\text{harmonics}}(\mathbf{X})]^{\mathsf{T}}$$

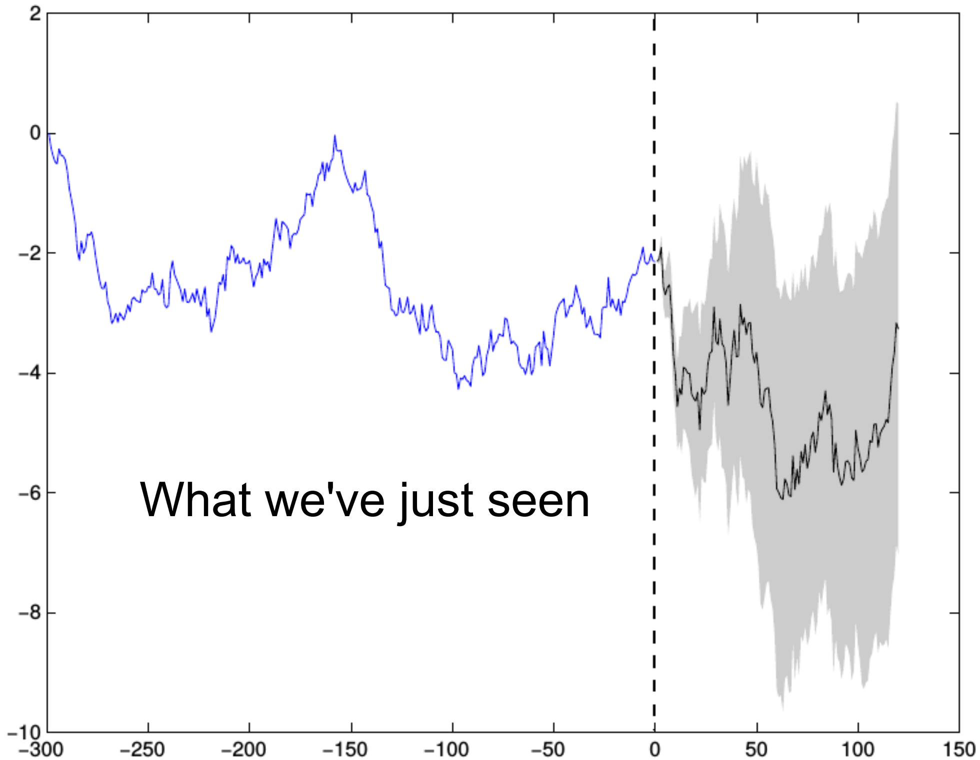


Method of analogues

A widely used method, especially in early **weather forecasting**
The following is the core assumption

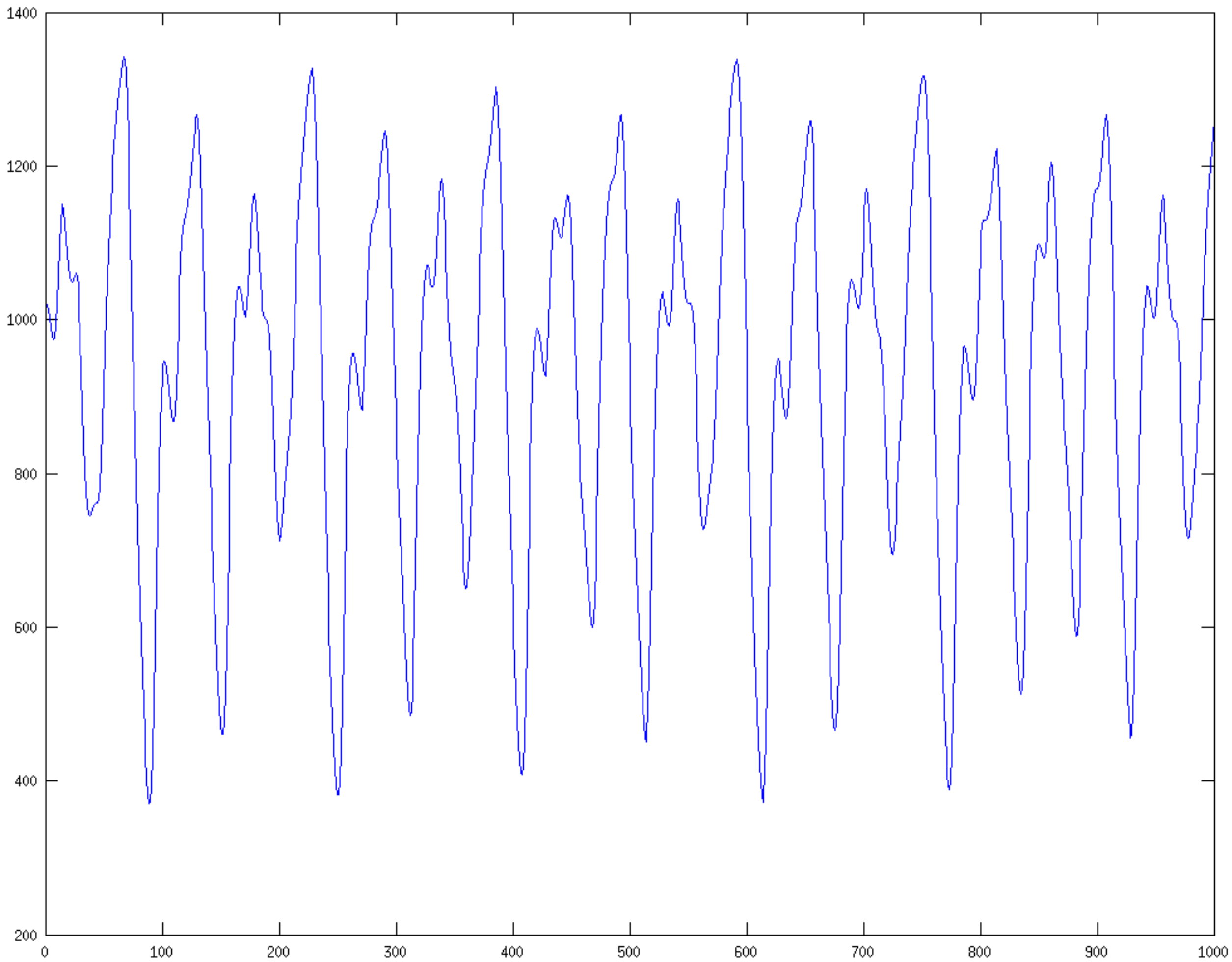
If the **recent past** of a time series, is similar to historical sequences we have **previously seen** then the **future** will be similar to the 'futures' of those similar historic timeseries





$$X = \sum_i w_i x_i \qquad Y = \sum_i w_i y_i.$$

Attractor distance

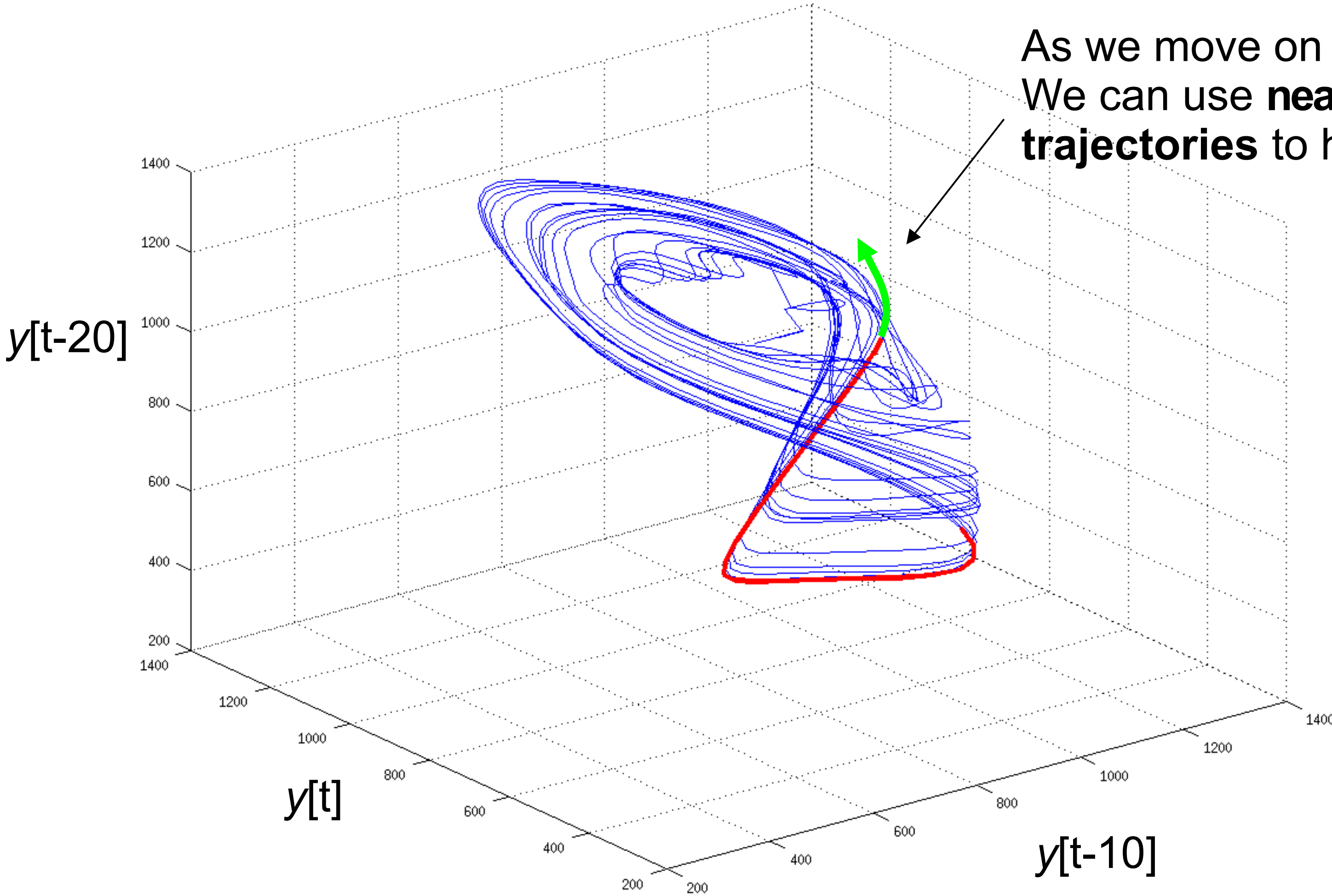


Mackey-Glass chaotic system

Method of embedding

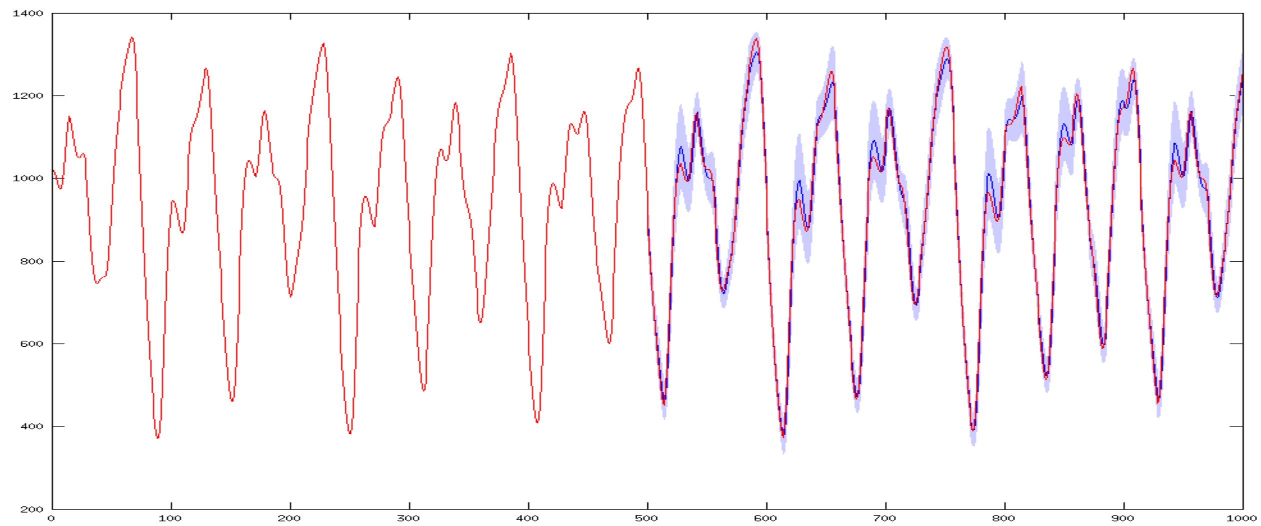
(Takens) – we can reconstruct the attractor of a dynamical system using a **tapped delay line** – i.e. **lagged versions of the time series**

As we move on the attractor
We can use **nearby trajectories** to help forecast

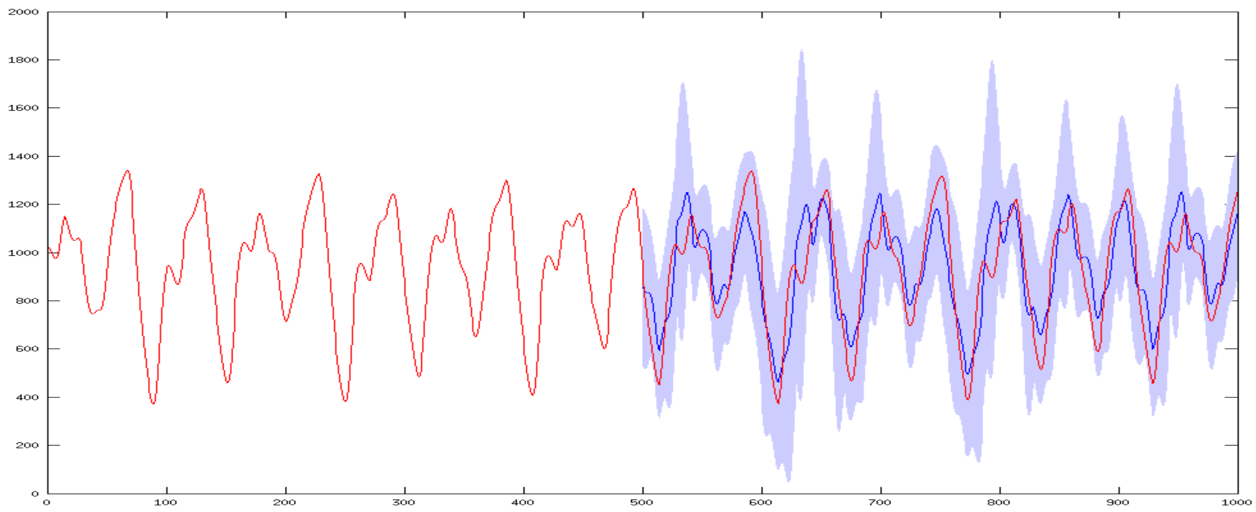


Improves performance

Using nearby trajectories



Using recent samples



Function Mappings - summary

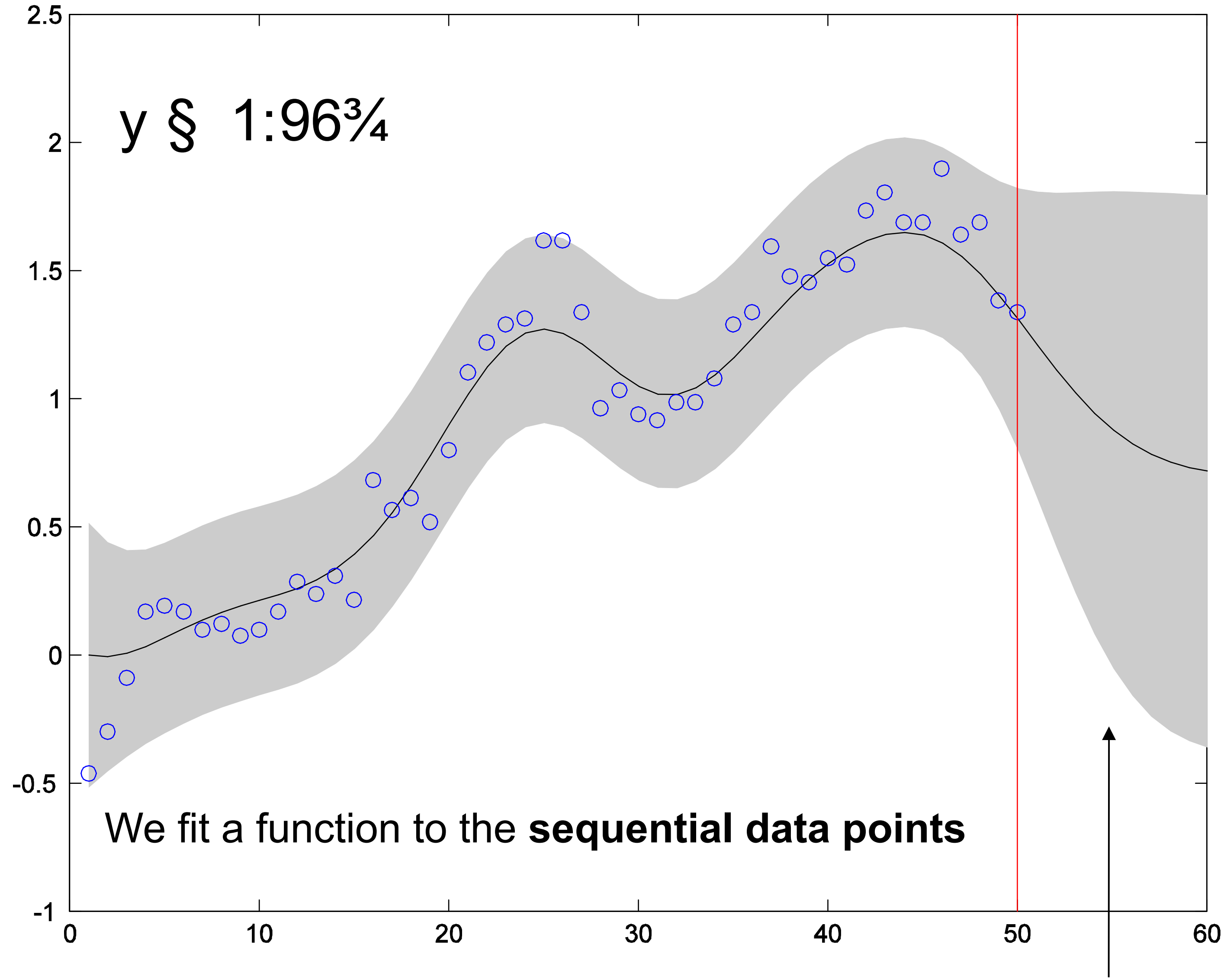
These are **widely used** and **very flexible**. We run the risk of moving from *timeseries* to *machine learning* – and of course, there is a vast overlap

The one problem of function mapping is that, without a lot of sophistication, the mapping we learn is **fixed**. Some ways around that

1) rolling window – estimate a mapping using a rolling **subset of the data**
2) adaptive models – for example the **Kalman filter**

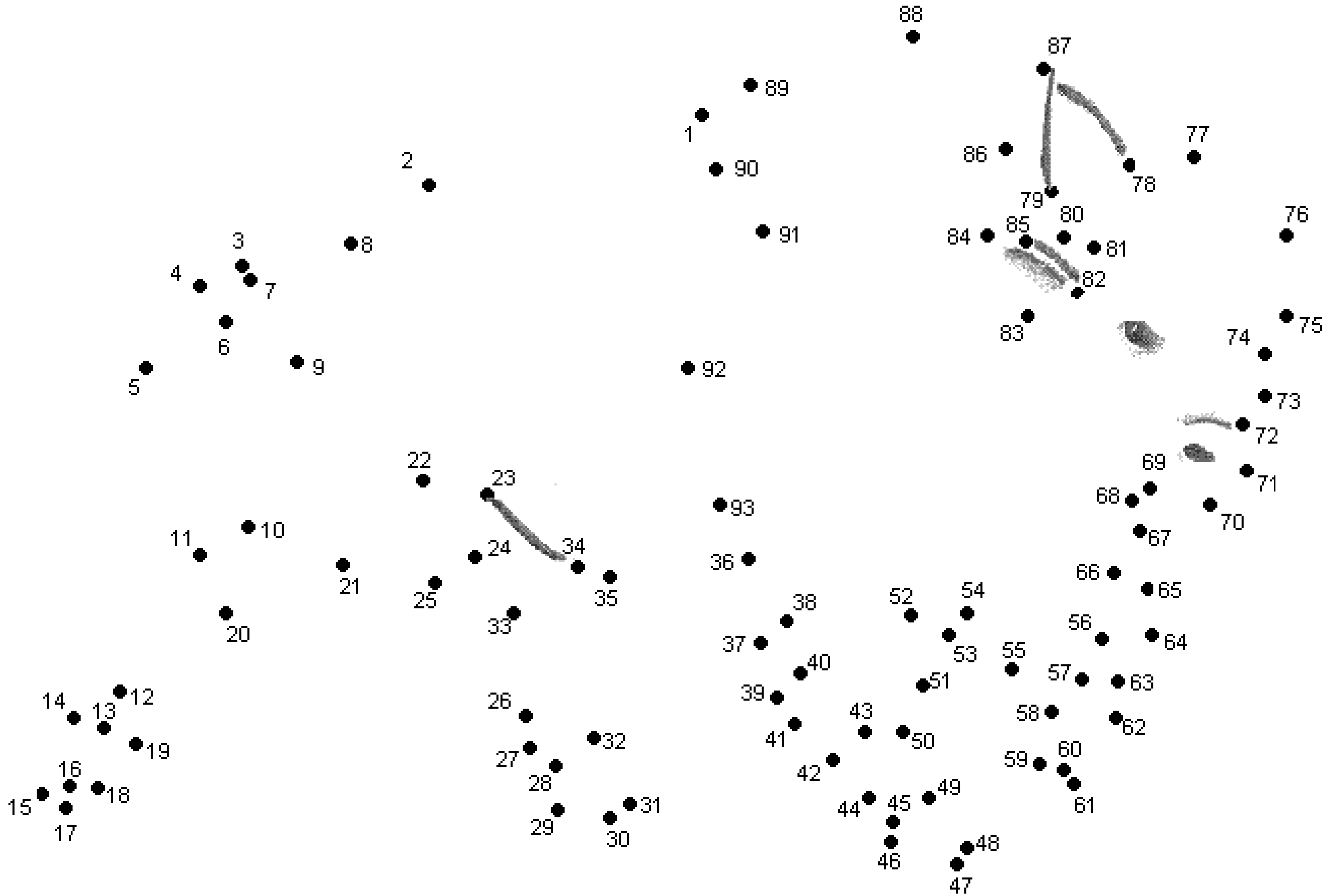
But now, let's go back though to the second prediction approach – that of **curve fitting.** Here we regress a function **through the time-varying values of the time series** and **extrapolate** (or **interpolate** if we want to fill in **missing values**) in order to **predict**

Curve fitting – is regression

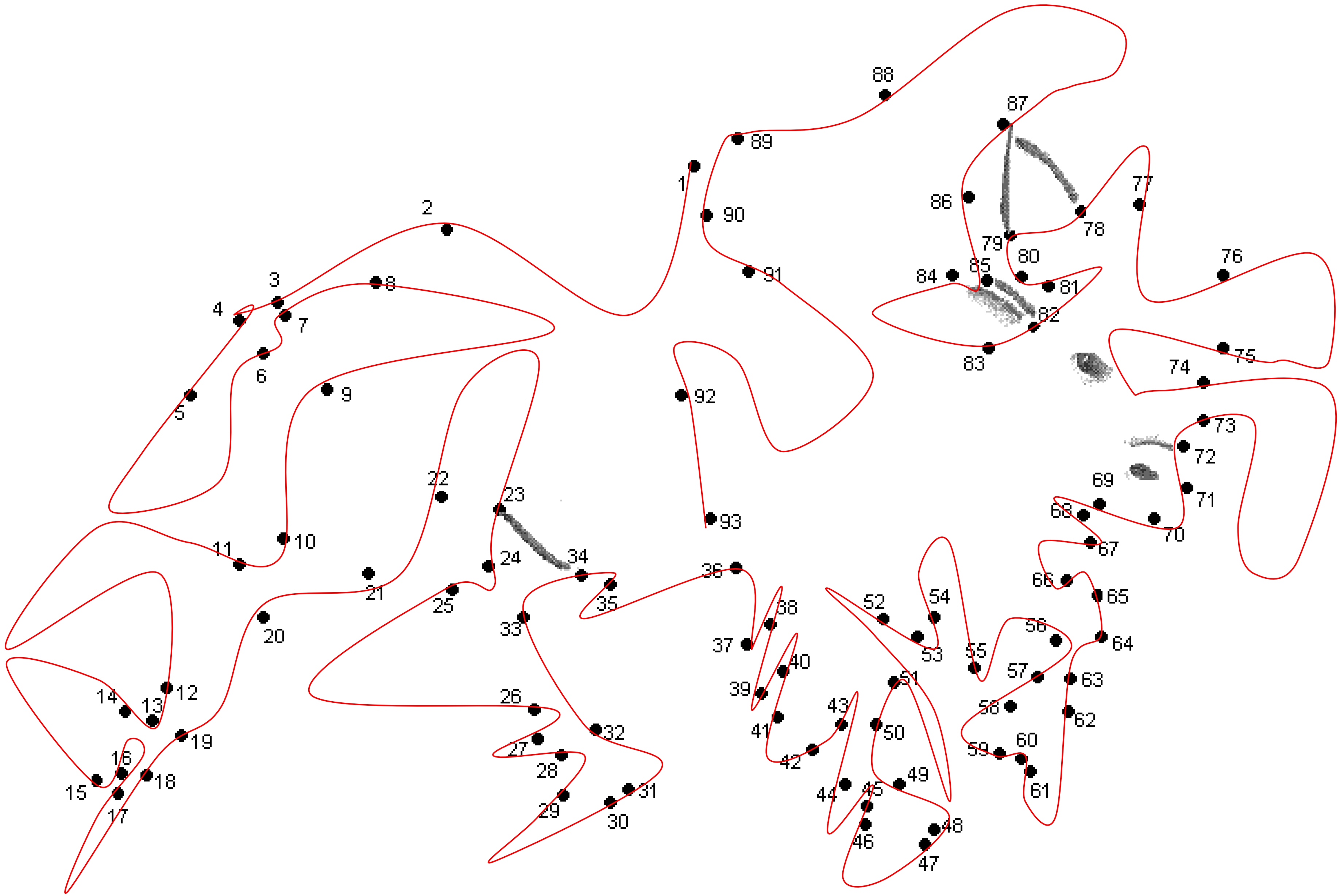


then **extrapolate the function**

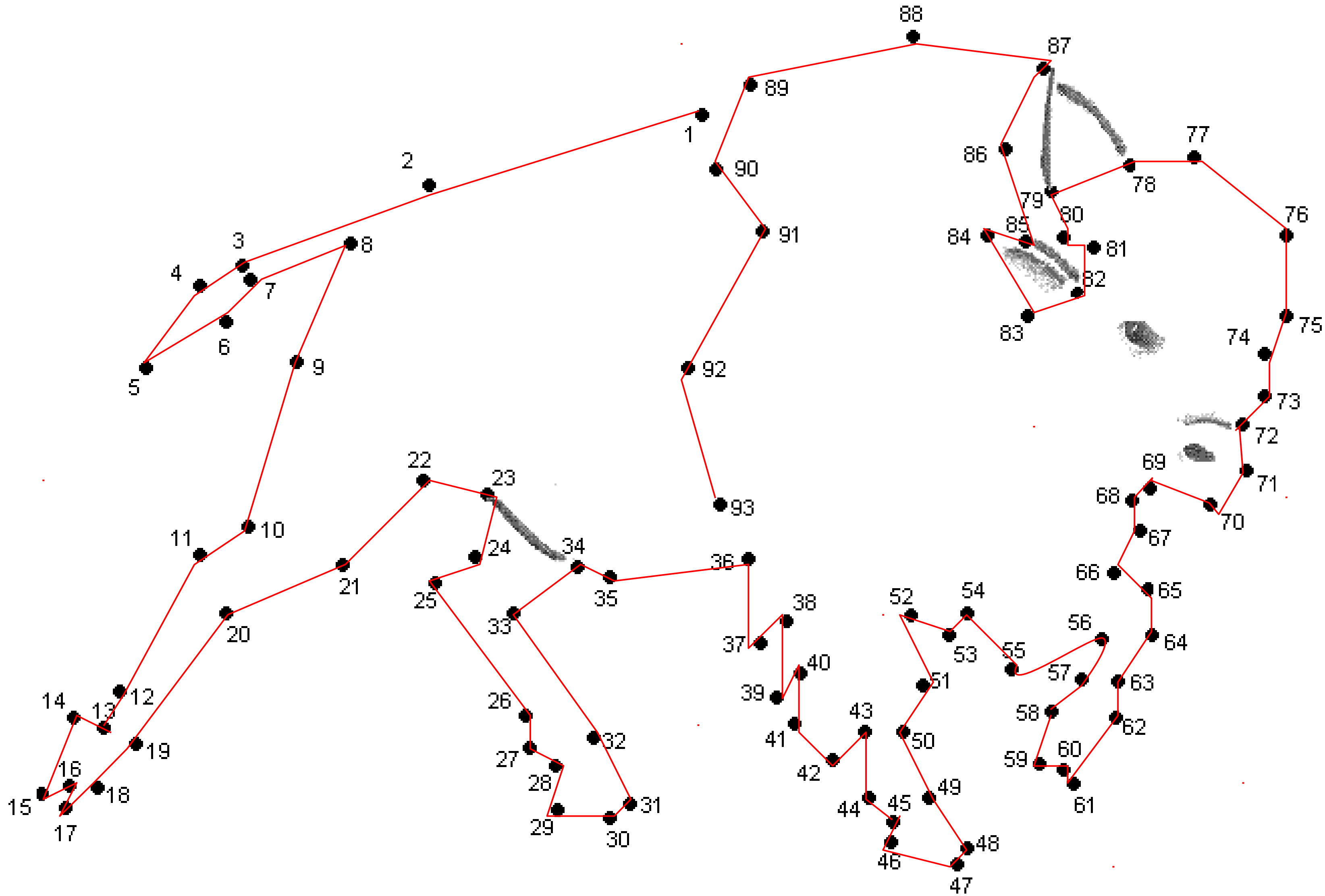
But... what form should the curve take?



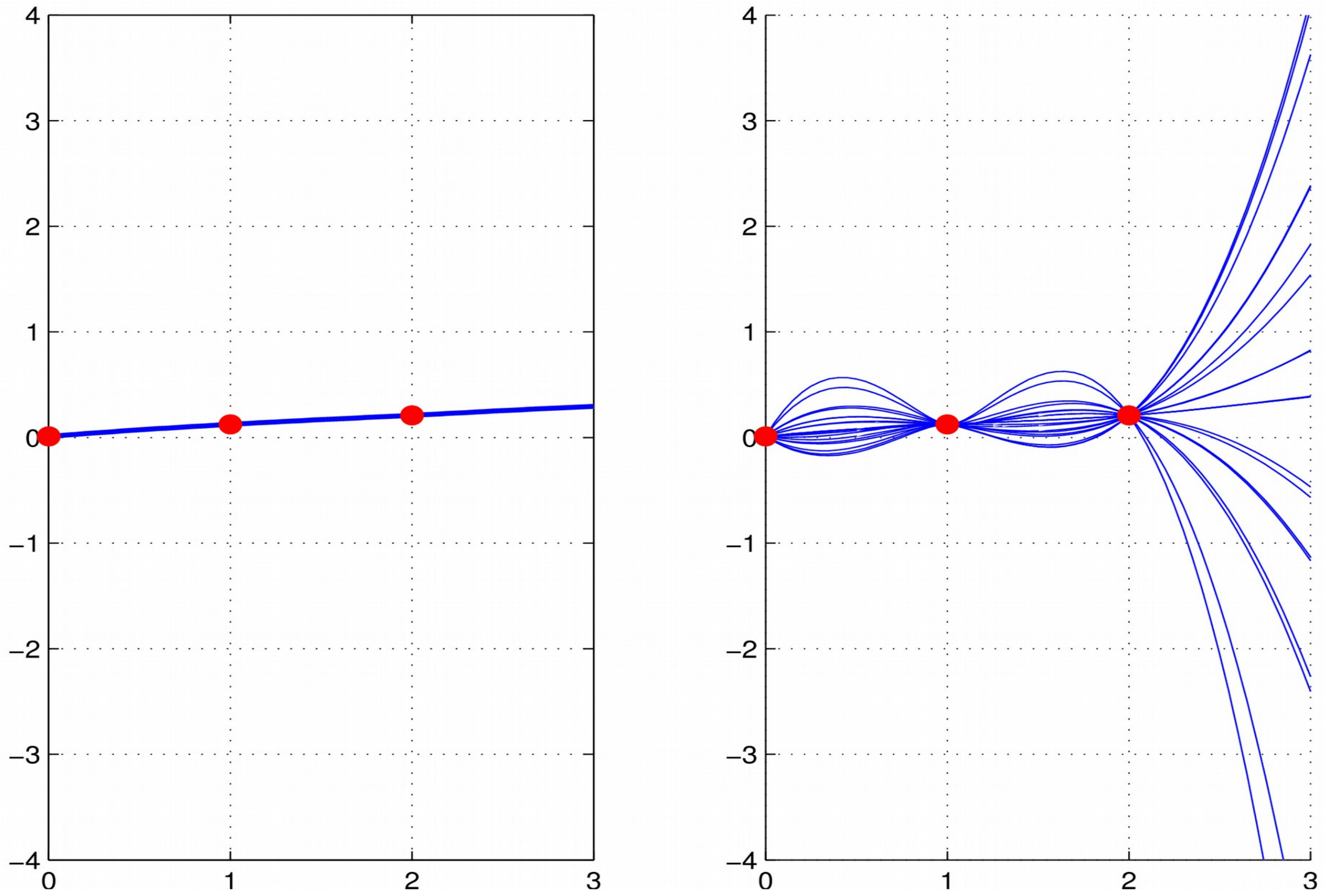
...this is a problem with many possible solutions



Prior information may allow us to discriminate between solutions

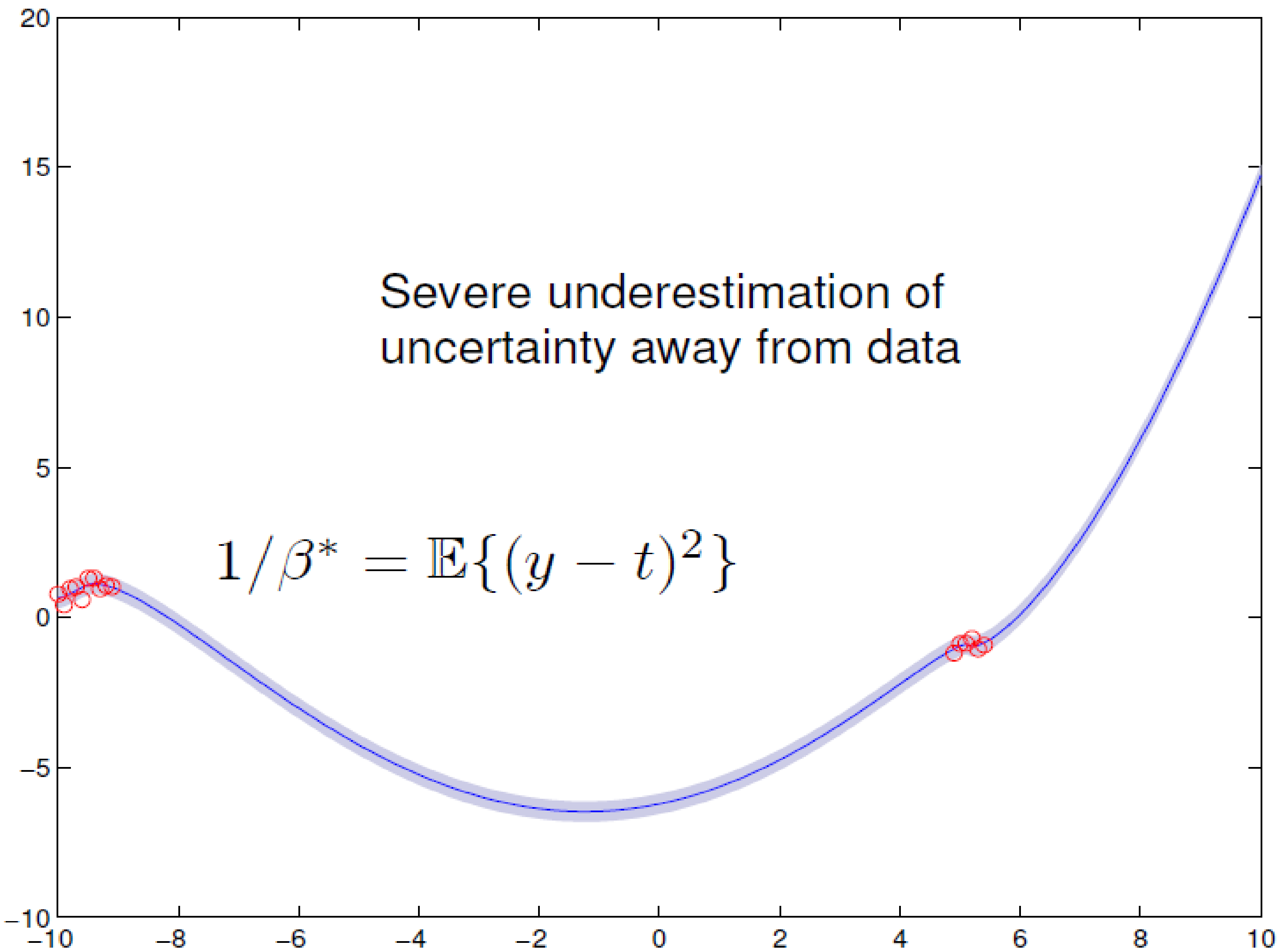


The right model?

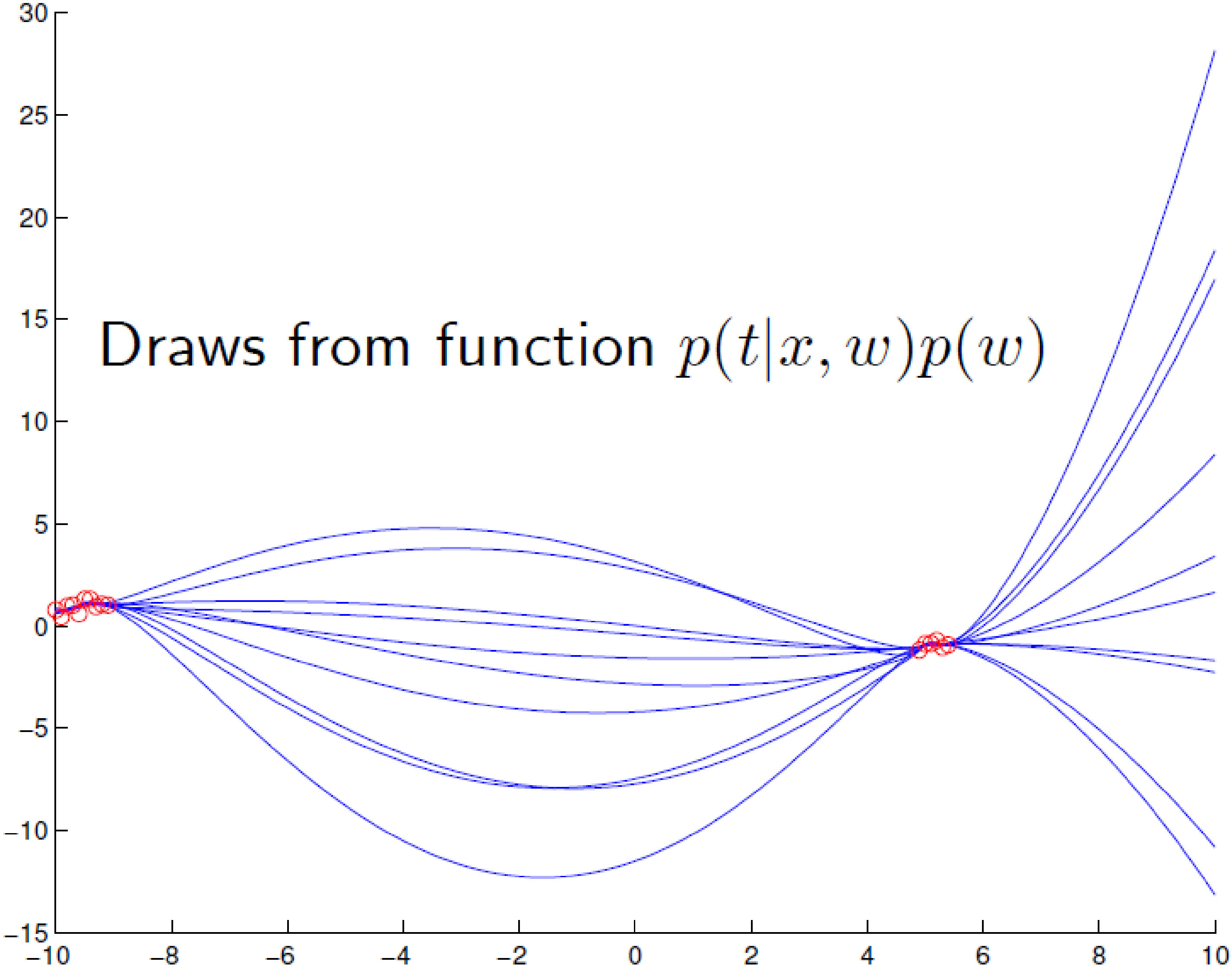


All these models explain the data equally well...

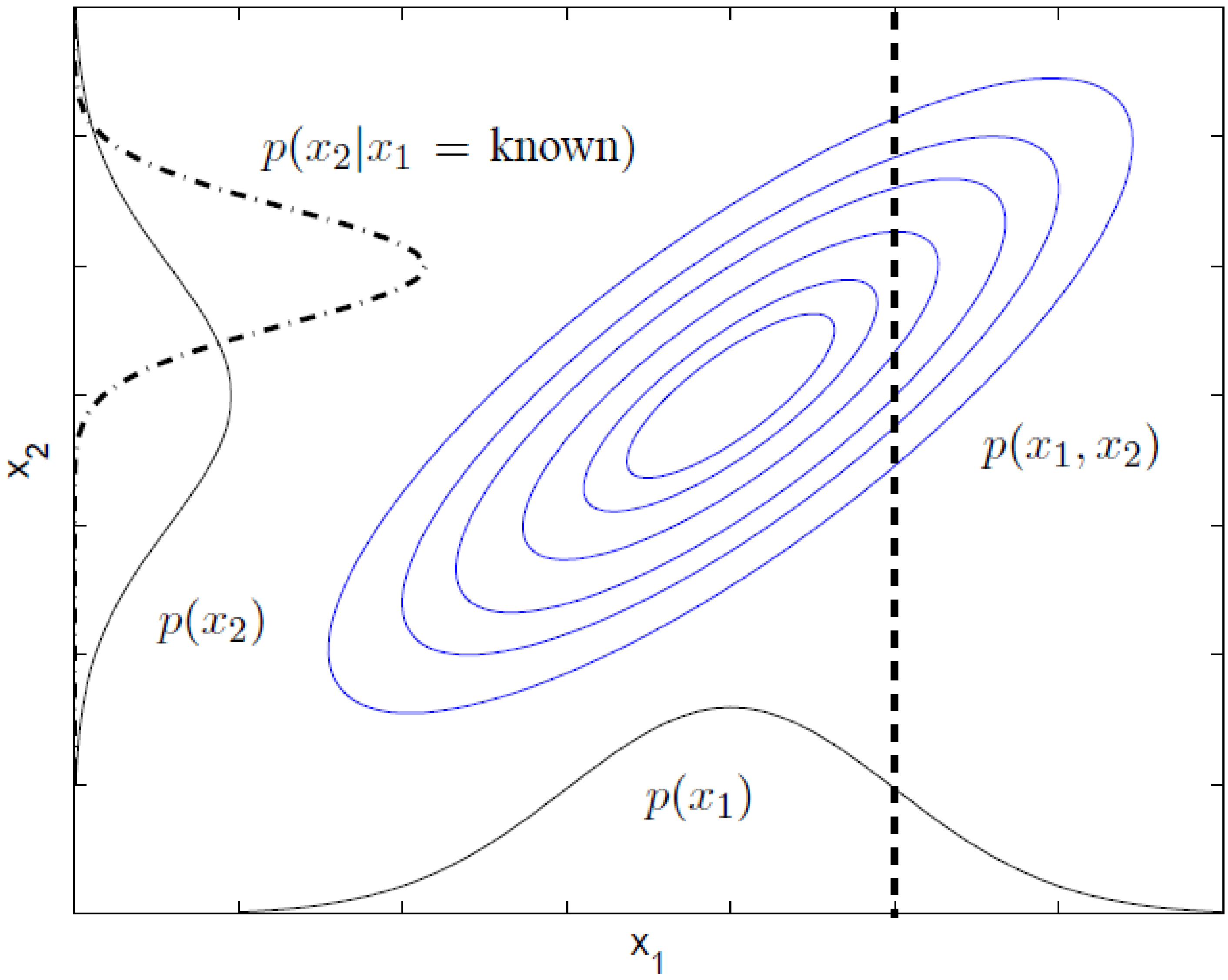
Maximum-likelihood solution

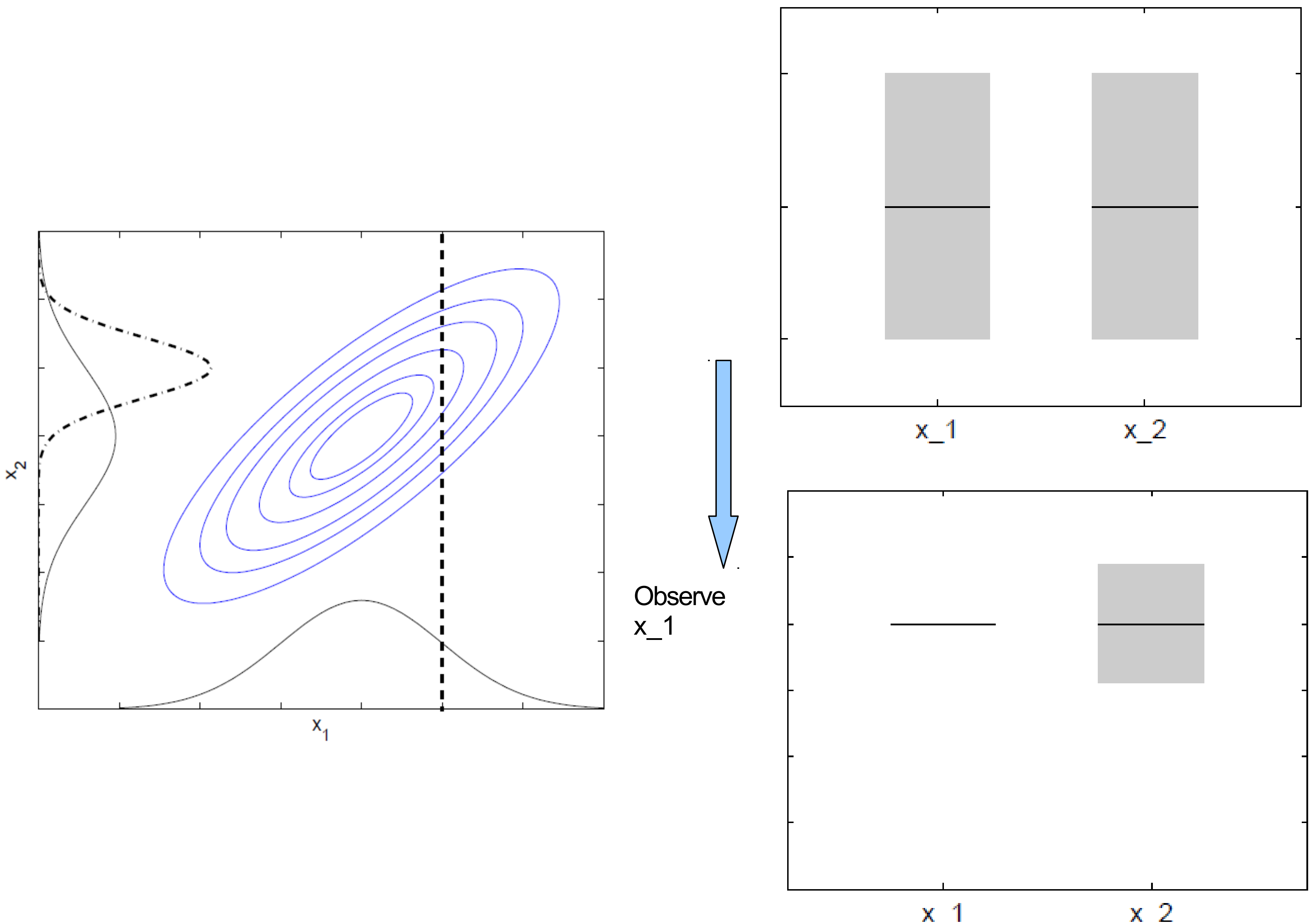


Draws from posterior



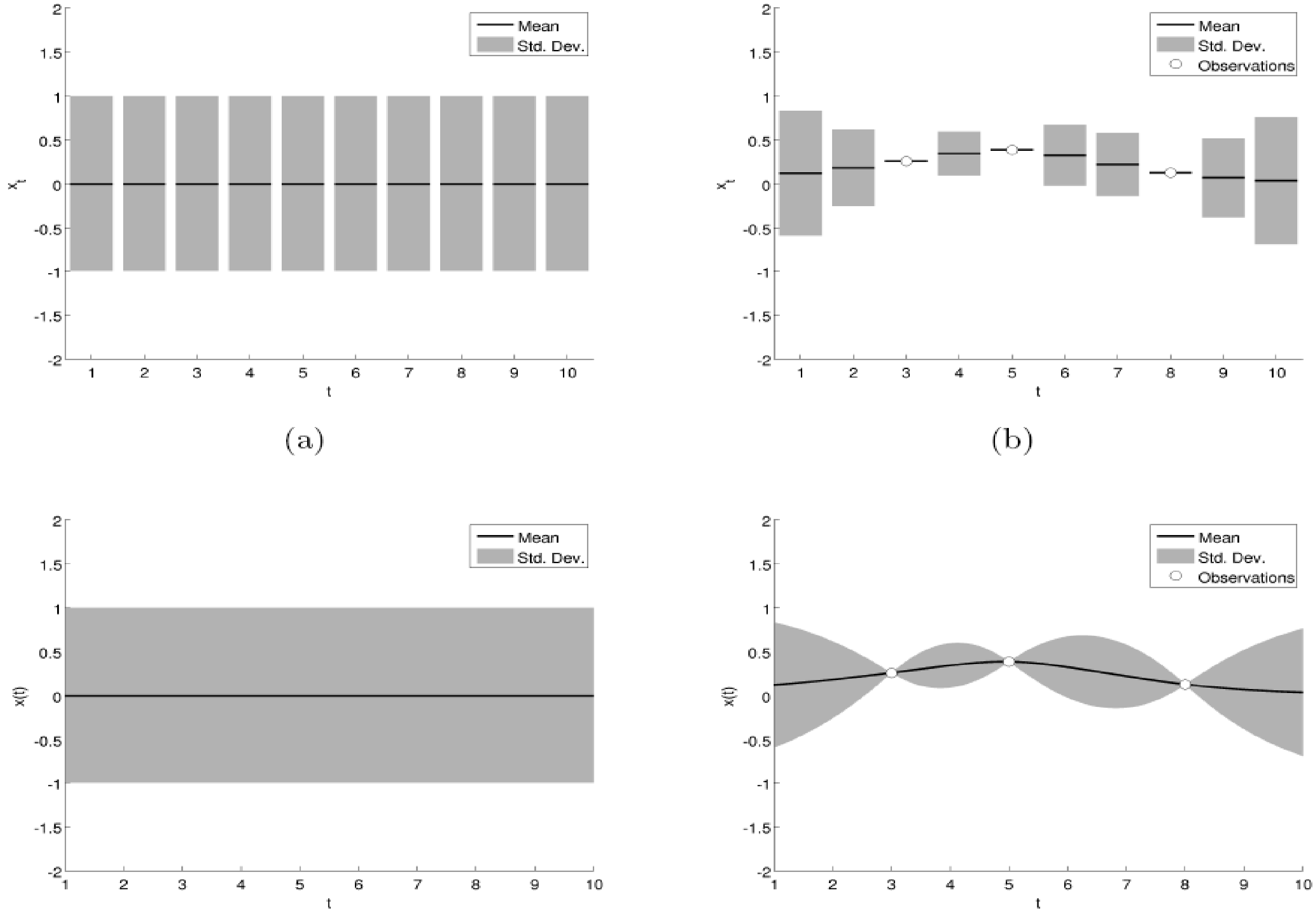
The humble (but useful) Gaussian





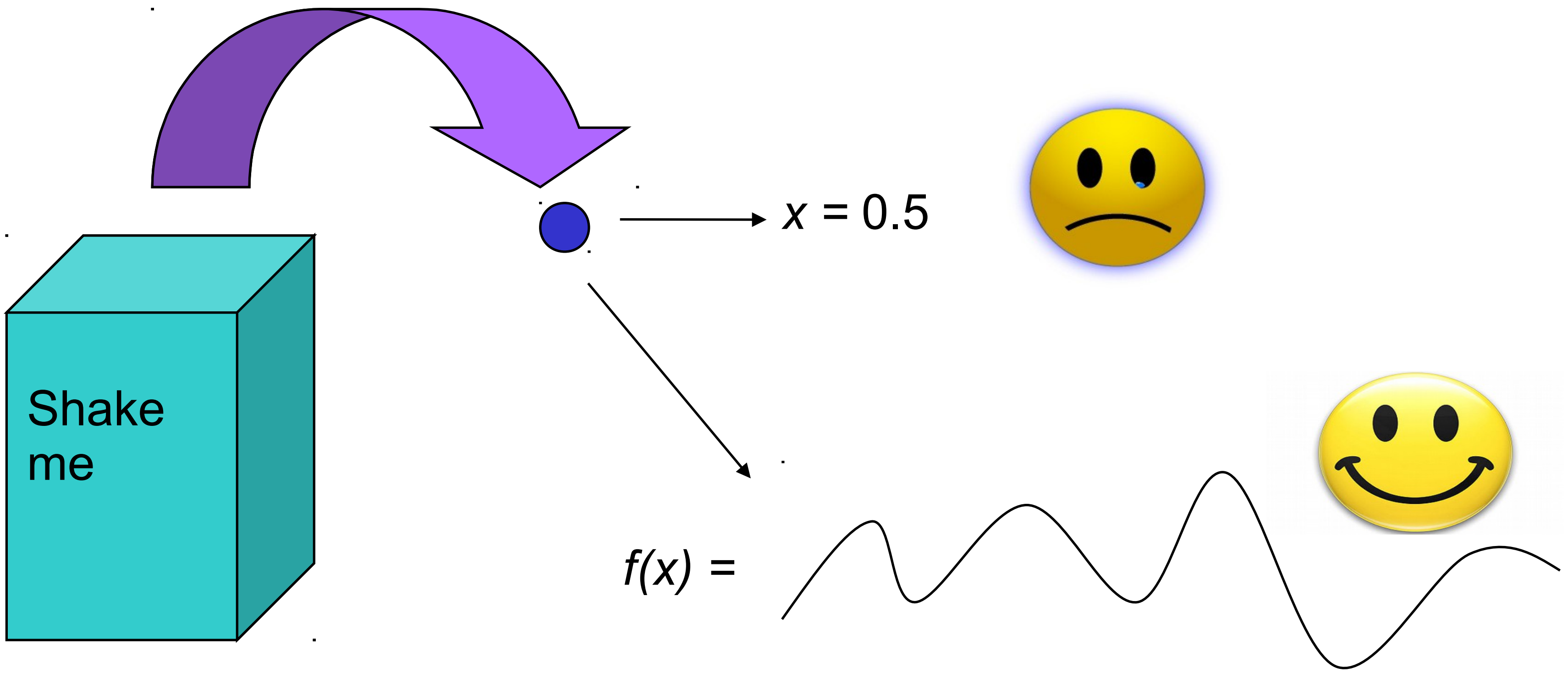
x1
x2
Observe
x_1
x_1
x_2
x_1
x_2

Extend to continuous variable



(a) (b) (c) (d)

Probabilities over functions not samples



A “X” *process* is a distribution over a function space such that the pdf at any evaluation of the function are conditionally “X” distributed.

-Dirichlet Process [infinite state HMM]

-Indian Buffet Process [infinite binary strings] etc etc.

The Gaussian process model

- See the GP via the distribution

$$\boxed{p(\mathbf{y}(\mathbf{x})) = \mathcal{N}(\boldsymbol{\mu}(\mathbf{x}), \mathbf{K}(\mathbf{x}, \mathbf{x}))}$$

- If we observe a set (x,y) and want to infer *y** at *x**

$$p\left(\begin{bmatrix} \mathbf{y} \\ y_* \end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}(\mathbf{x}) \\ \mu(x_*) \end{bmatrix}, \begin{bmatrix} \mathbf{K}(\mathbf{x}, \mathbf{x}) & \mathbf{K}(\mathbf{x}, x_*) \\ \mathbf{K}(x_*, \mathbf{x}) & k(x_*, x_*) \end{bmatrix}\right)$$

$$p(\mathbf{y}_*) = \mathcal{N}(\mathbf{m}_*, \mathbf{C}_*)$$

$$m_* = \mu(x_*) + \mathbf{K}(x_*, \mathbf{x})\mathbf{K}(\mathbf{x}, \mathbf{x})^{-1}(\mathbf{y} - \boldsymbol{\mu}(\mathbf{x})),$$

$$\sigma_*^2 = K(x_*, x_*) - \mathbf{K}(x_*, \mathbf{x})\mathbf{K}(\mathbf{x}, \mathbf{x})^{-1}\mathbf{K}(\mathbf{x}, x_*).$$

The beating heart...

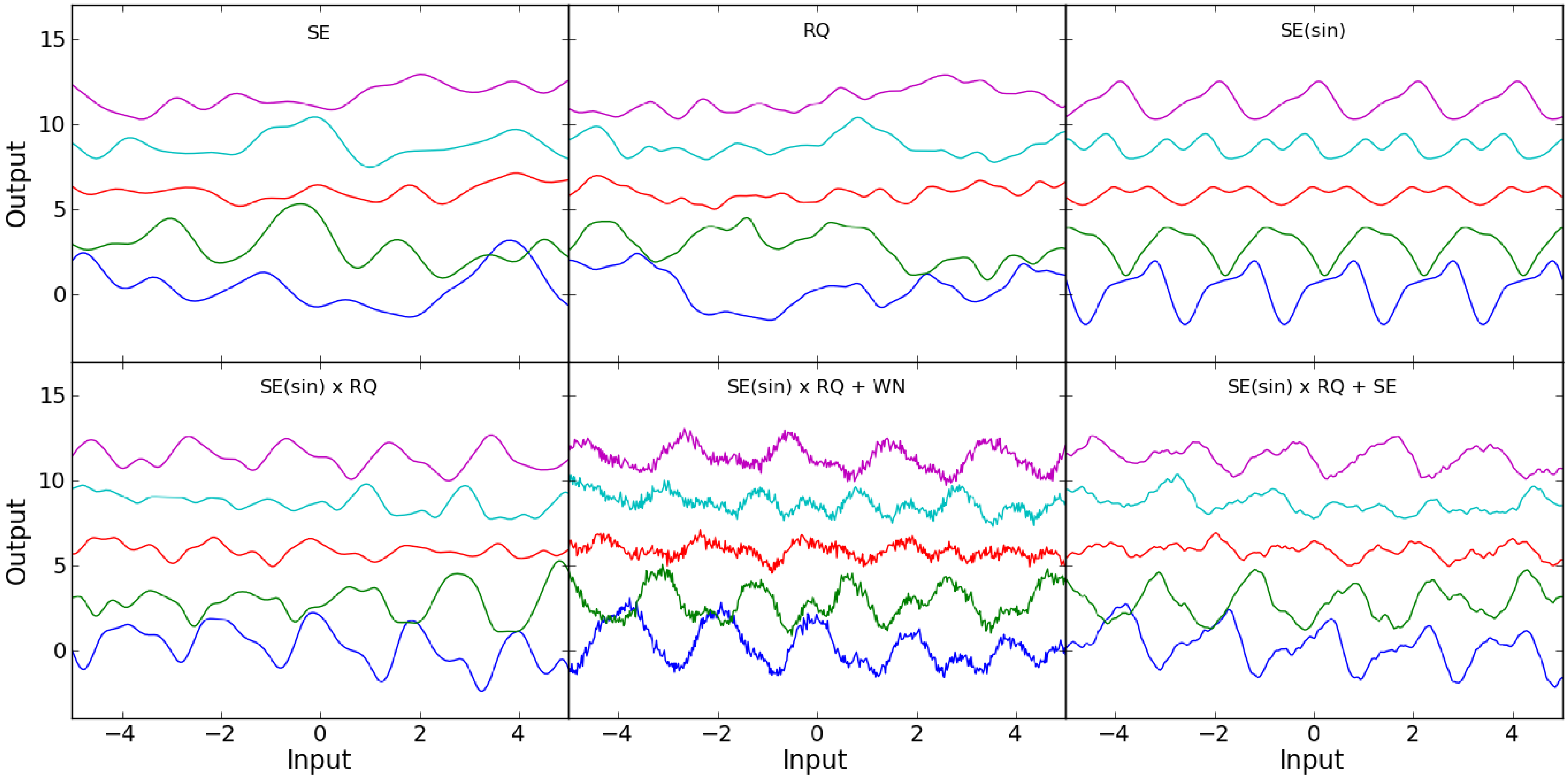
What about these covariances though?

$$\mathbf{K}(\mathbf{x},\mathbf{x}) = \begin{pmatrix} k(x_1,x_1) & k(x_1,x_2) & \cdots & k(x_1,x_n) \\ k(x_2,x_1) & k(x_2,x_2) & \cdots & k(x_2,x_n) \\ \vdots & \vdots & \vdots & \vdots \\ k(x_n,x_1) & k(x_n,x_2) & \cdots & k(x_n,x_n) \end{pmatrix}$$

Achieved using a *kernel function*, which describes the relationship between two points

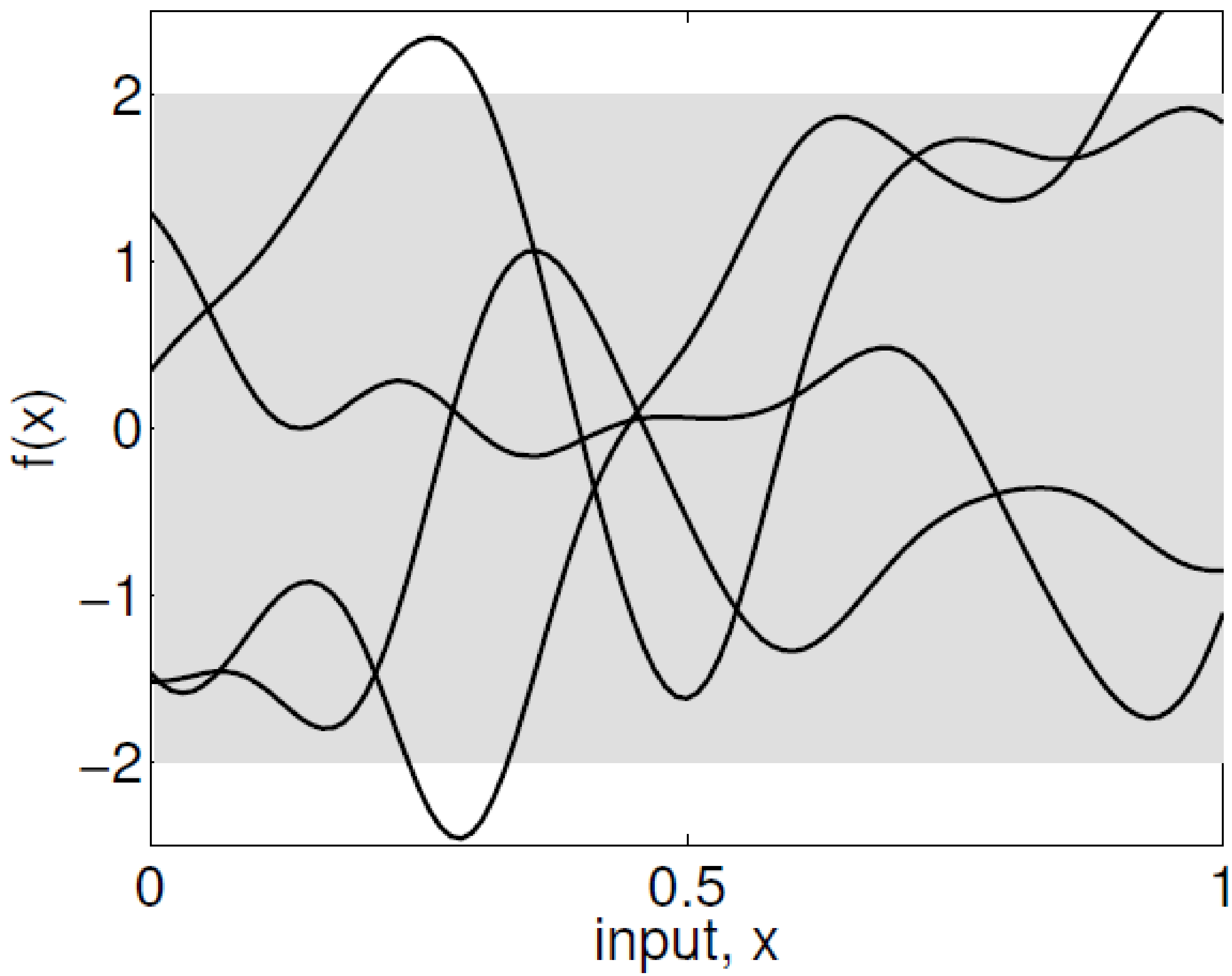
What form should this take though?

Kernel functions

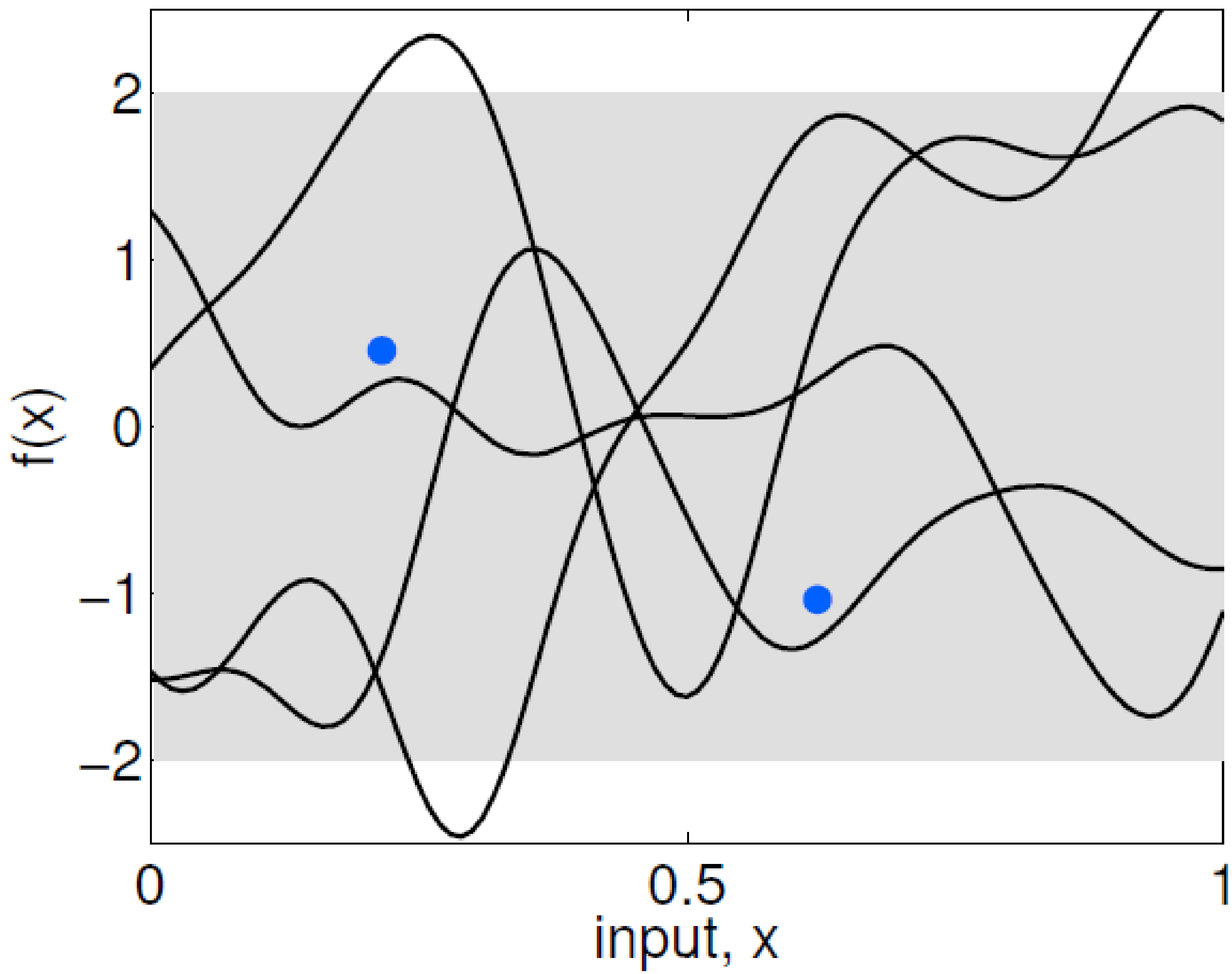


S. Roberts, M. Osborne, M. Ebden, S. Reece, N. Gibson and S. Aigrain (2012). Gaussian Processes for Timeseries Modelling Philosophical Transactions of the Royal Society (Part A).

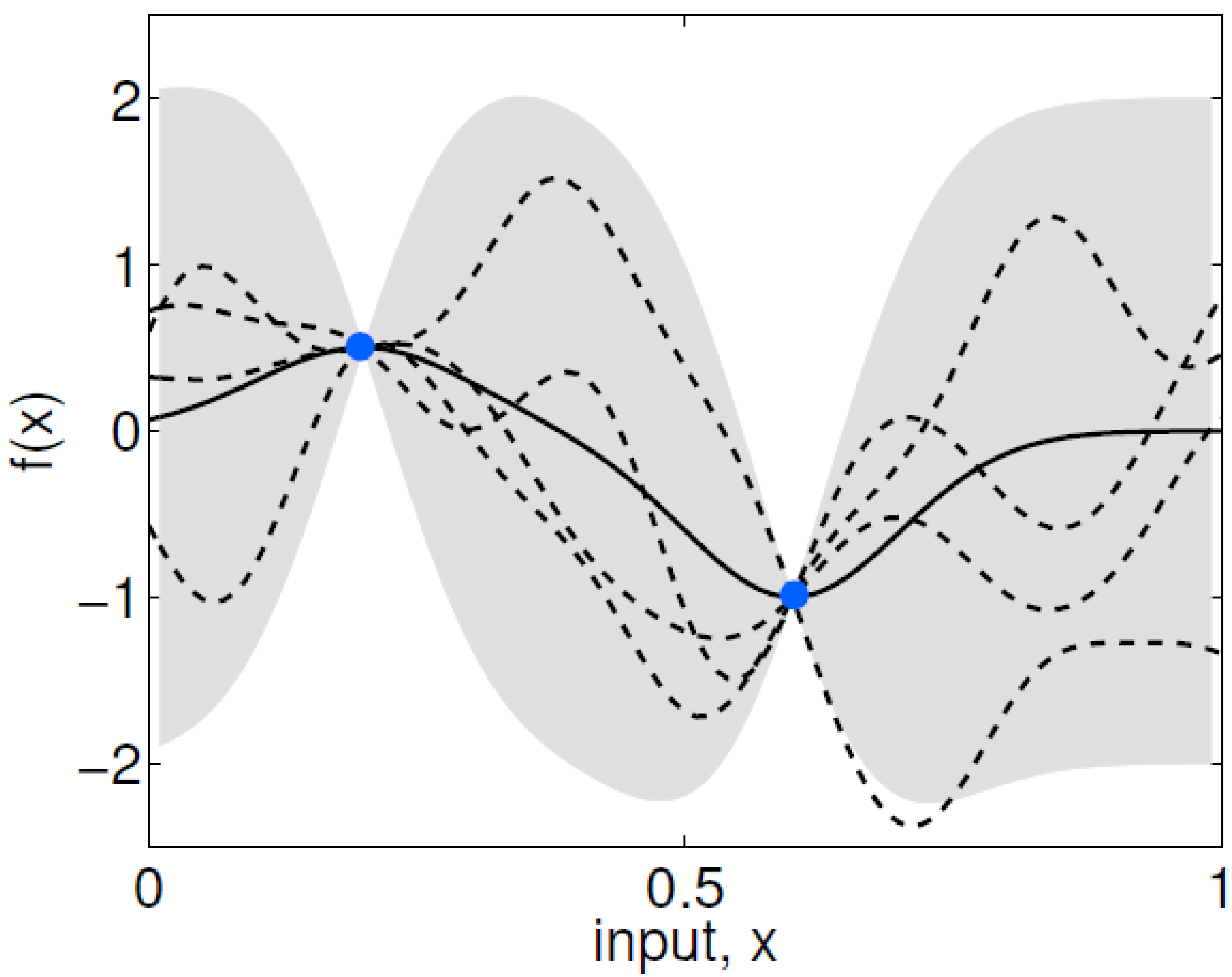
The GP experience



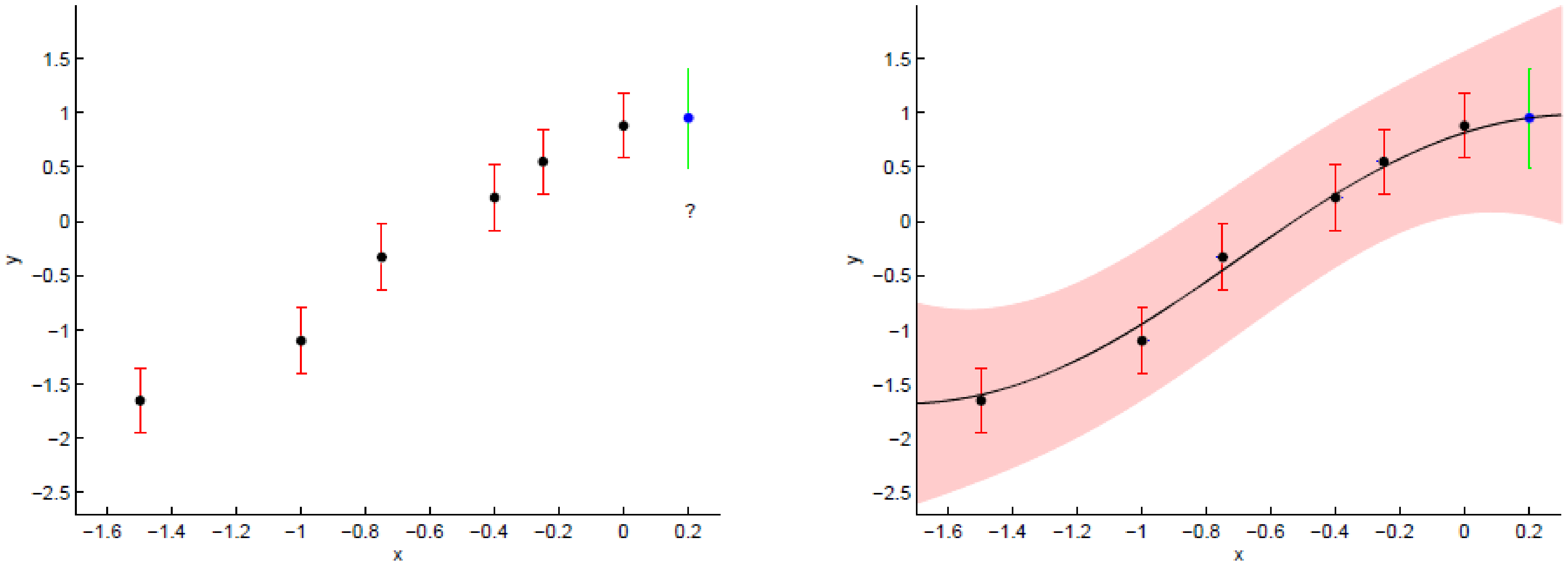
Observe some data



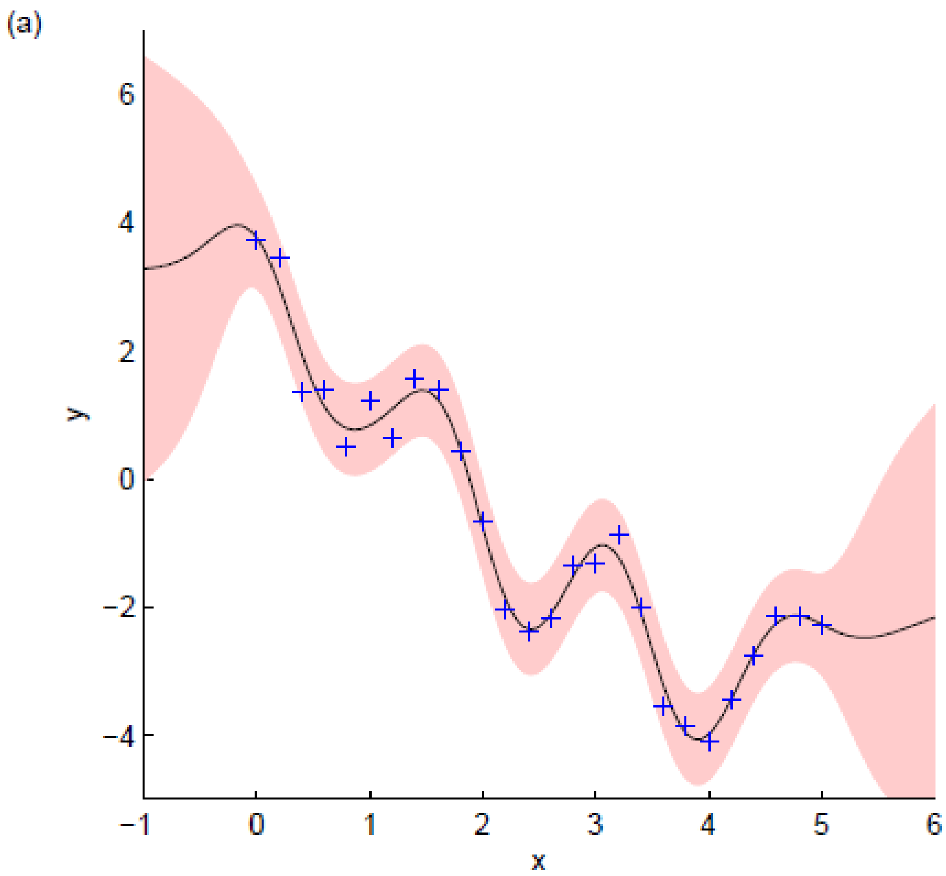
Condition posterior functions on data

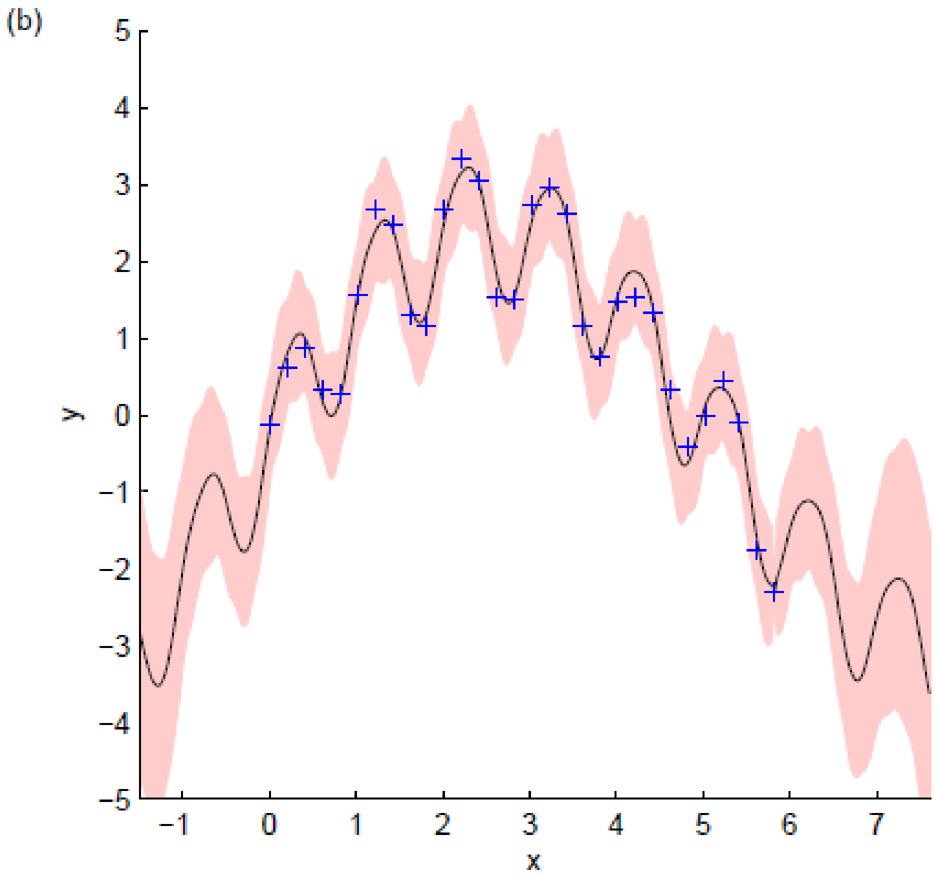


Simple regression modelling

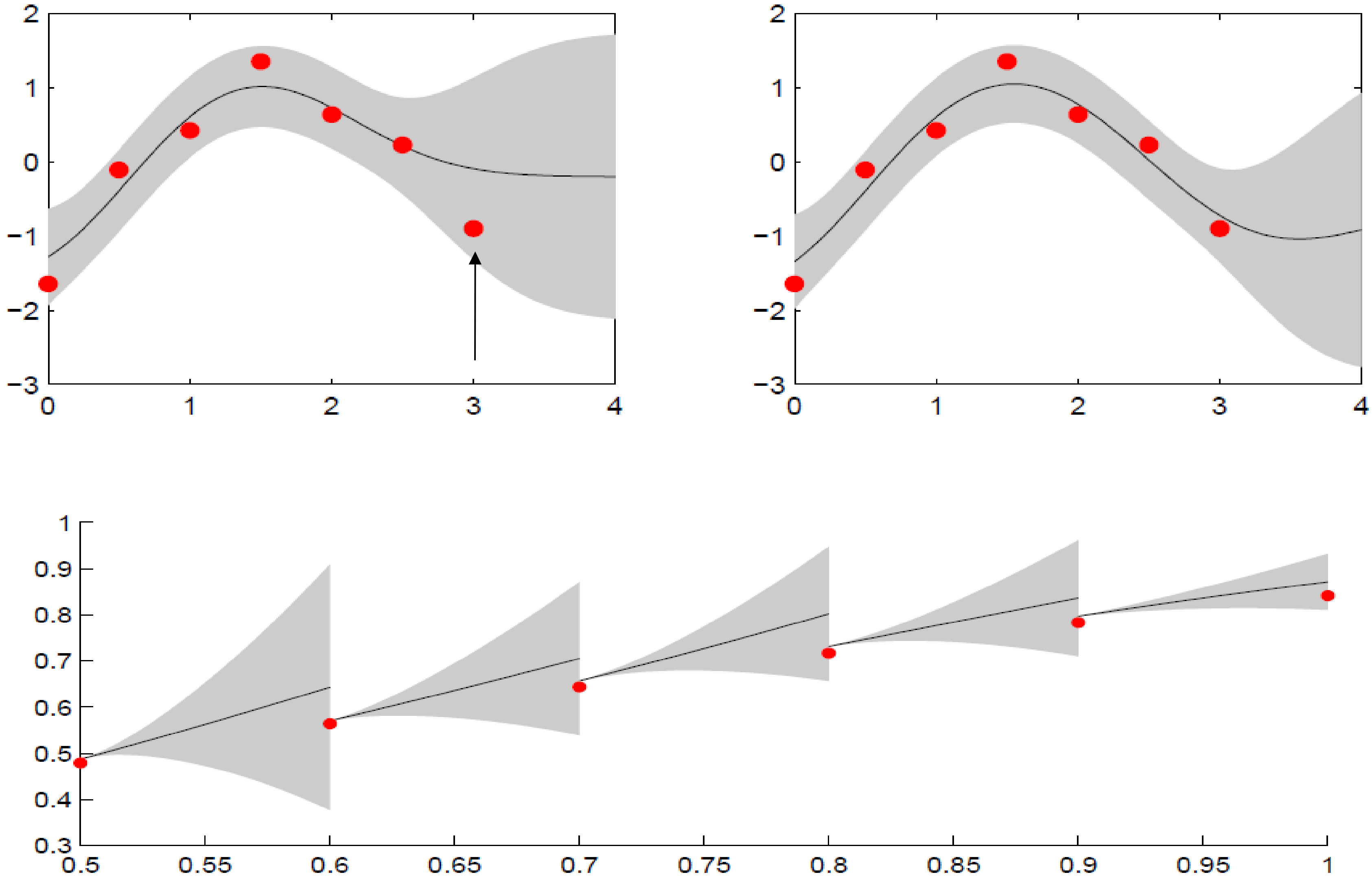


Less simple regression

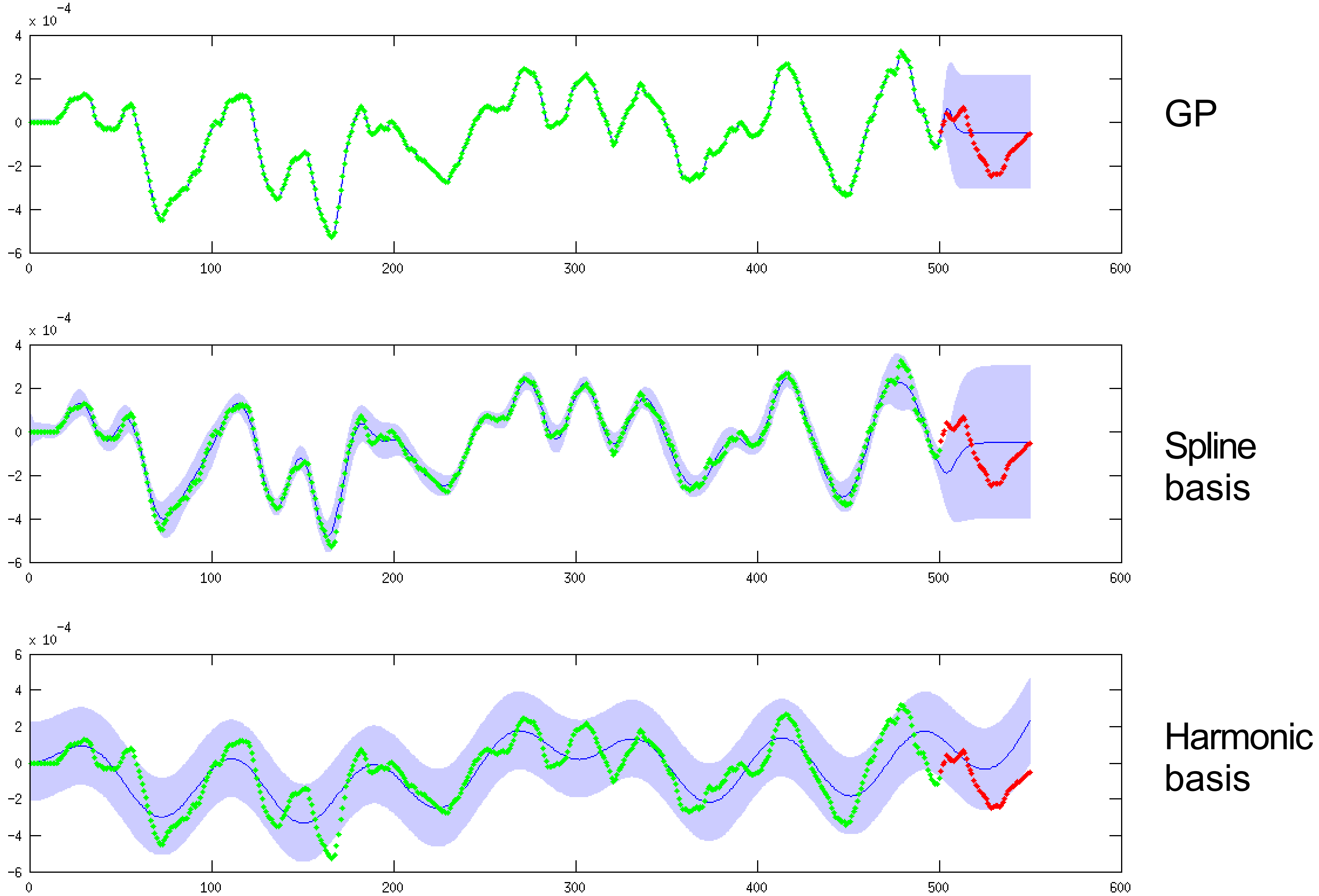




In a sequential setting



Simple comparison



The Kalman process revisited

In previous lectures we've seen how the Kalman updates produce an **optimal filter** under assumptions of **linearity and Gaussian noise**

The Kalman process is one of an **adaptive linear model**, so if we regress from **non-linear representation of the data** then it becomes easy to develop a **non-linear, adaptive model**

$$\hat{y}[t] = \sum_{n} w_n[t]\phi_n(Y_{past})$$

Coping with missing values

Missing observations in the data stream *y*[t]

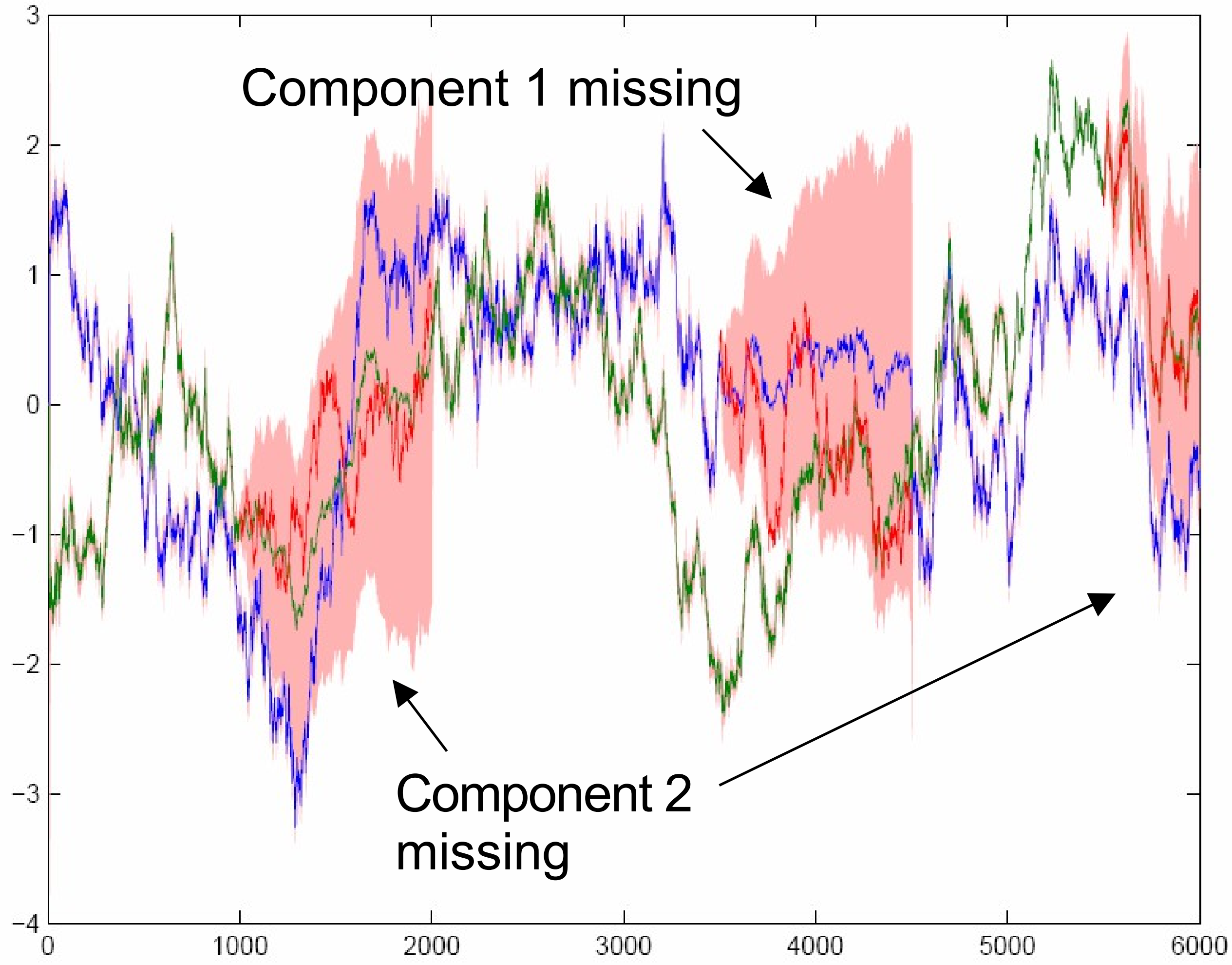
Can infer all or part of the missing observations vector as state-space model is *linear Gaussian* in the observations – simply replace the true observation with the inferred one.

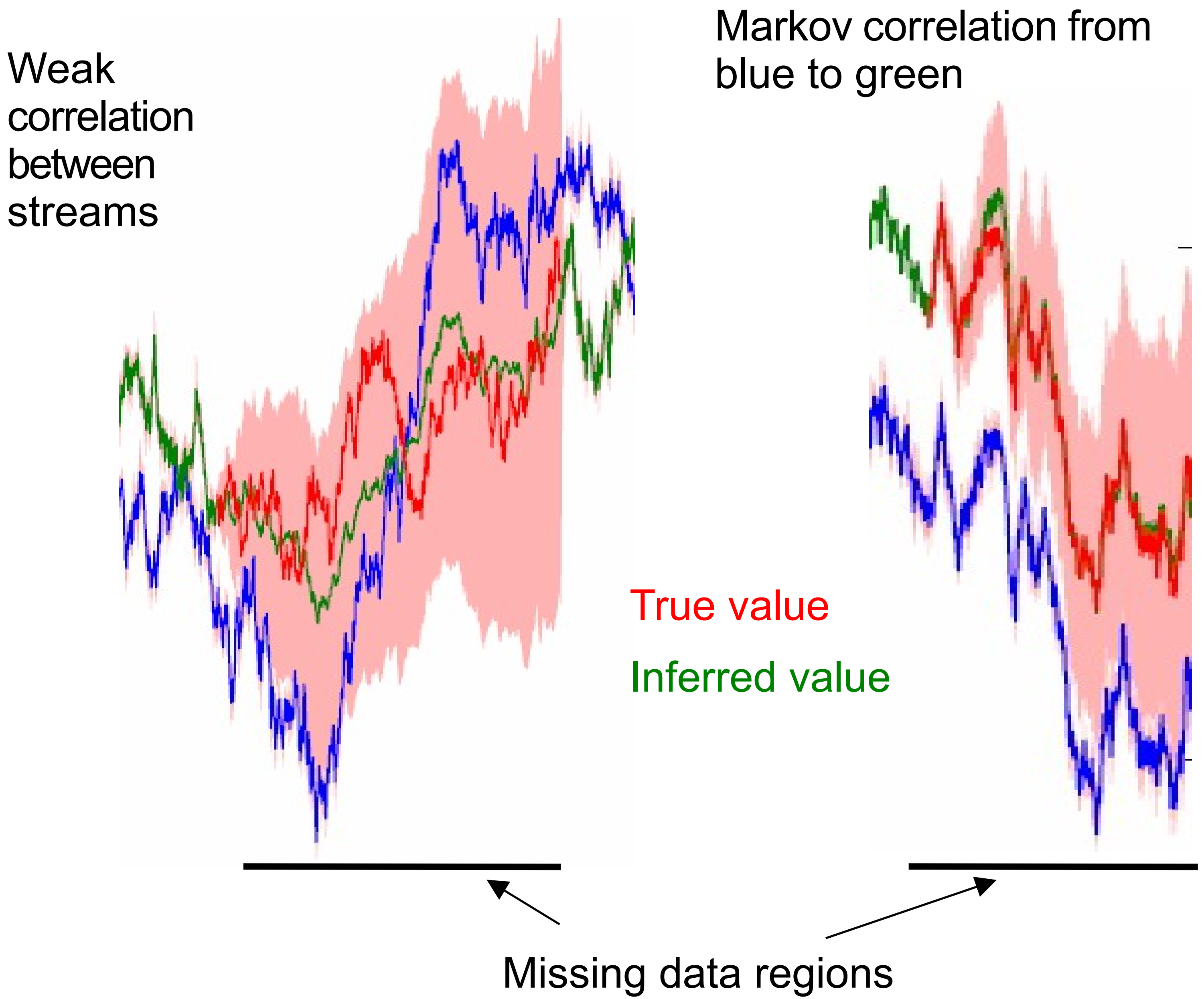
If the model is for time-series prediction, then proxy observations are simply the most probable posterior predictions from the past time steps – this naturally leads to a sequential AR process.

$$\boxed{\hat{y}[t] = \sum_n w_n[t]\tilde{y}[t-n]}$$

Could be directly observed or inferred

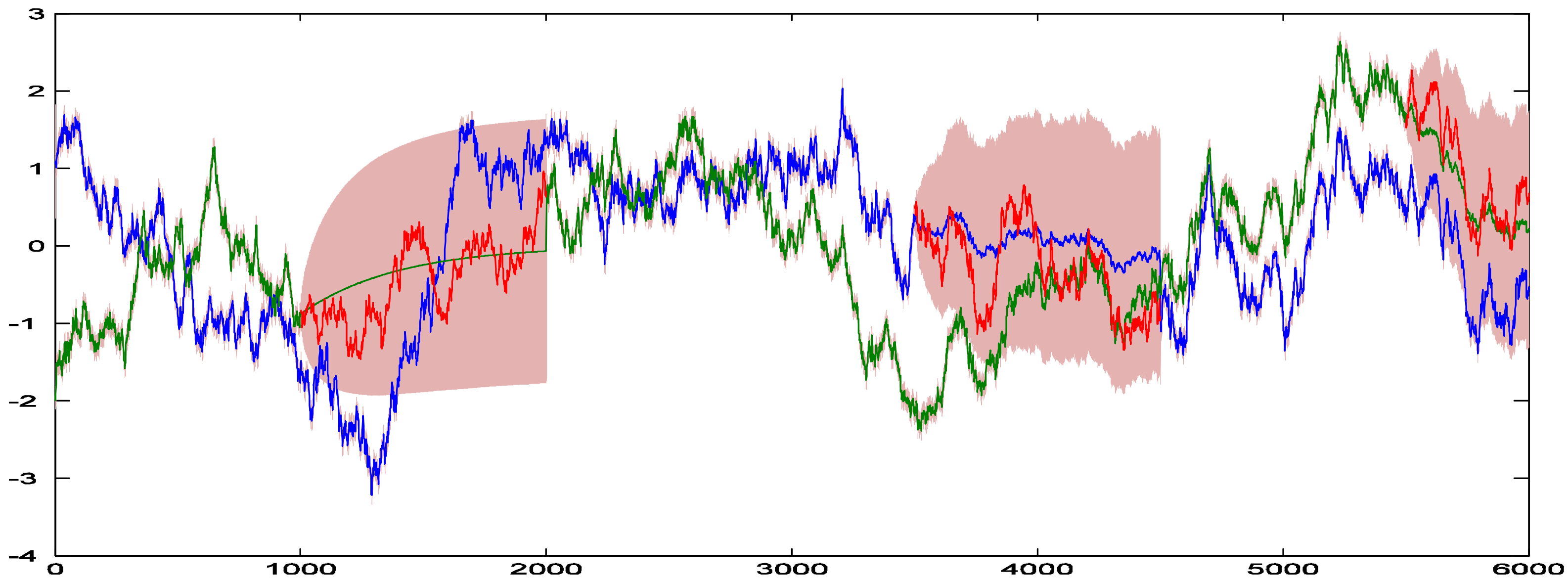
Brownian stream example



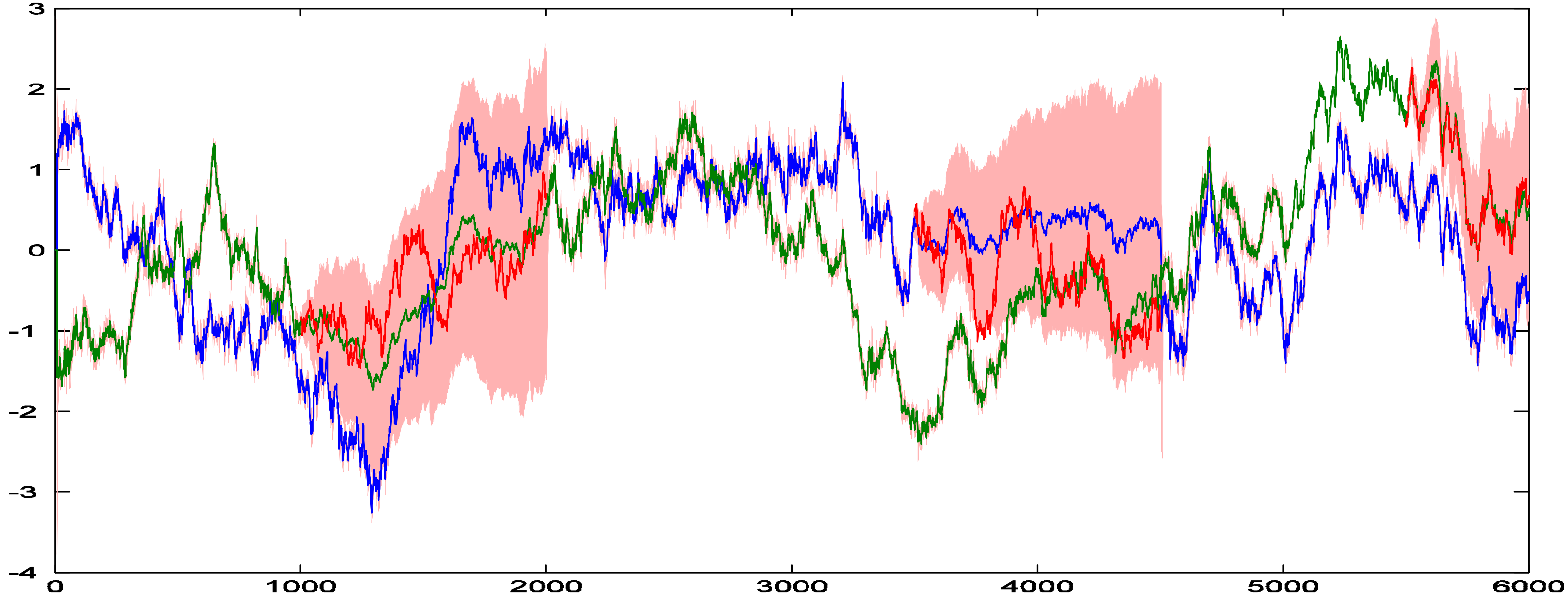

Weak correlation between streams
Markov correlation from blue to green
True value
Inferred value
Missing data regions

Comparison – synthetic Brownian data

GP



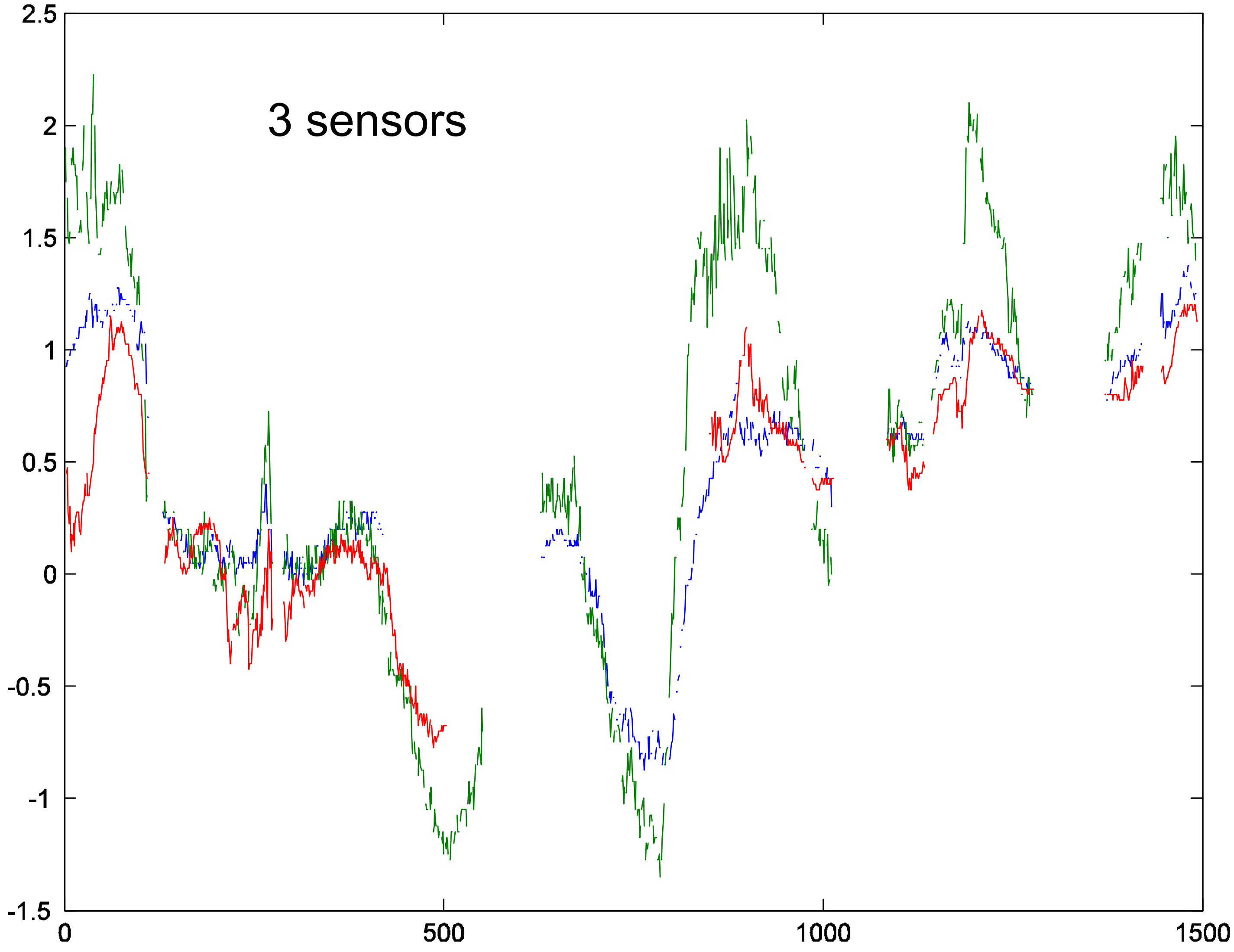
KF



Application example

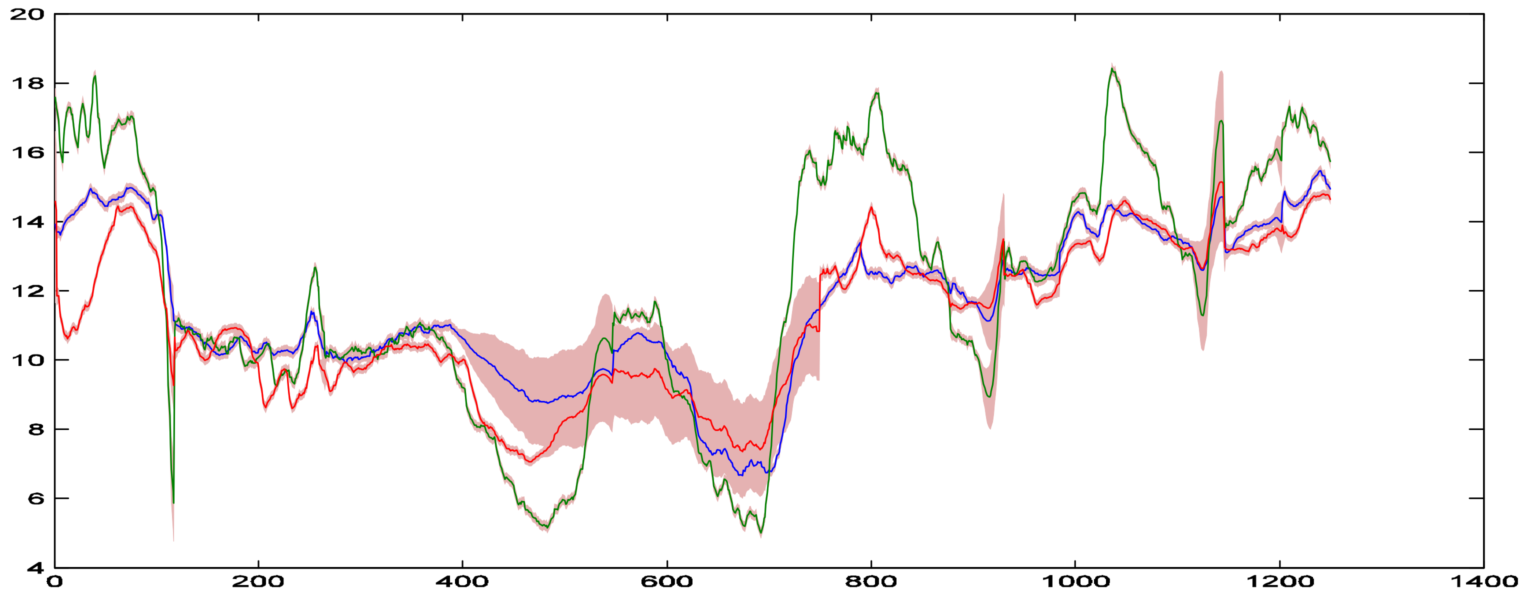
Set of weather stations – local weather information

Comparison – air temperature

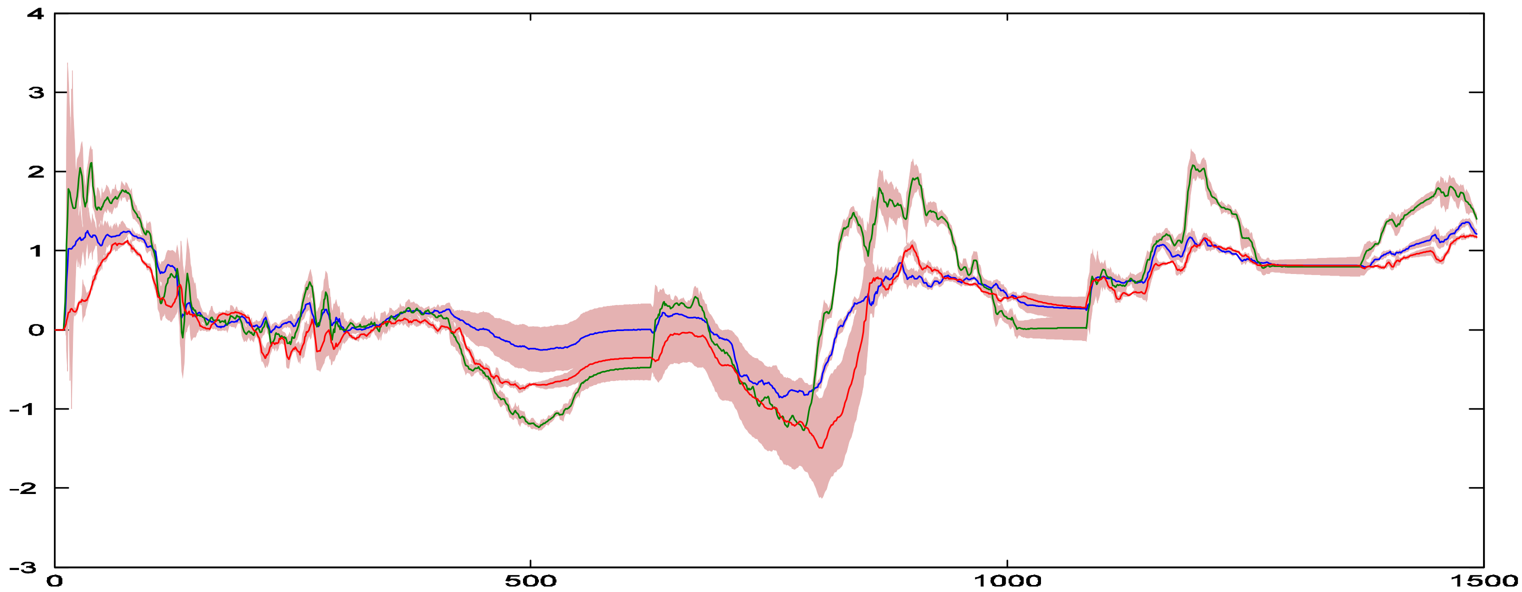


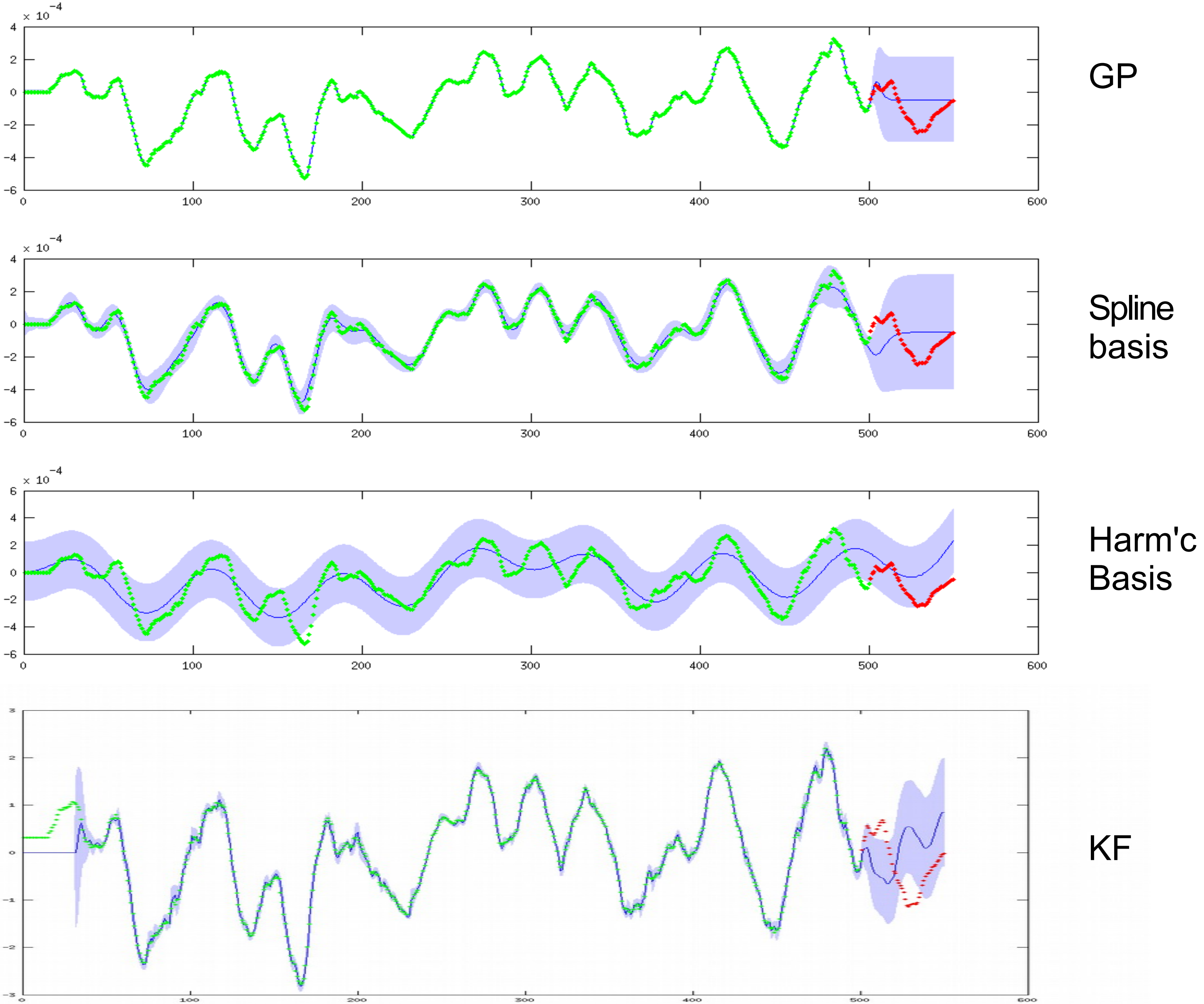
Air temperature

GP



KF




GP
Spline basis
Harm'c Basis
KF

Comparison : GP v KF

State Space models

Computationally very efficient

Infer posteriors over outcome variables

Handle missing data and corruptions at all levels

Can extend to sequential / predictive *decision* processes with ease

Active data requesting (request for observation or label)

Prior knowledge of data *dynamics*

Gaussian Processes

Computationally demanding, but satisfactory for real-time systems

Infer posteriors over all variables, including hyper-parameters

Handling missing data and corruptions at all levels

More difficult to extend to decision processes at present

Active data requesting

Prior knowledge regarding nature of data *correlation length*

Recent innovation sees intimate relationships between GPs and SSMs

Particle filtering

In much of what we have looked at so far, we have assumed that the **posterior distribution** has some simple form – for example it is **Gaussian**

All we then need to do is to infer the **posterior mean and (co-)variance**

Why is this assumption useful?

- it means we can readily map the **prior Gaussian to the posterior**

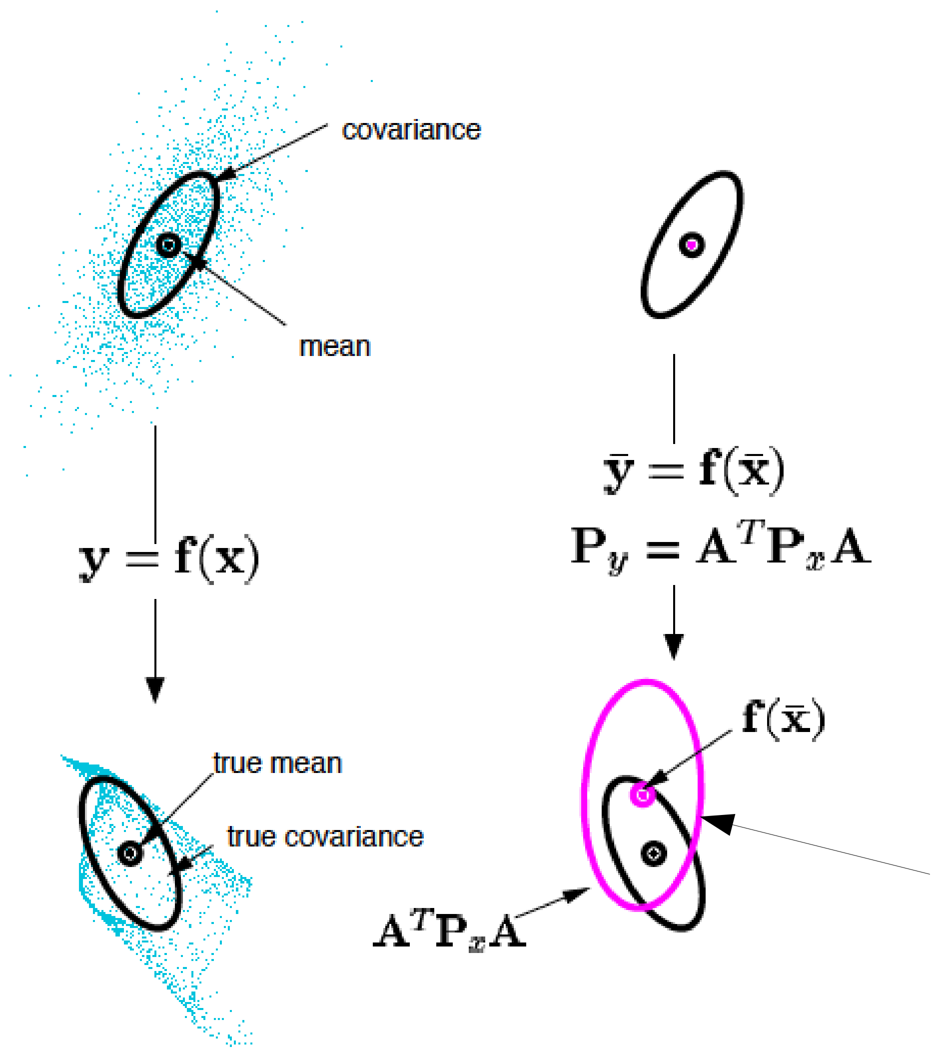
Many systems, though are not that simple – there may be **multi-modalities** and the posterior is **non-Gaussian**. Indeed it might even be that there is **no simple parametric model that describes it** (at least that we know about ahead of time)

Let's think about a simple system that shows that this Gaussian assumption fails

$$y[t] = y[t-1]^2$$

If $y[t-1]$ has a Gaussian posterior, used as prior to $y[t]$, then we know that the **prior cannot be conjugate with the posterior** as $y[t]$ **cannot be Gaussian distributed**

So what's wrong with the EKF?



EKF solution is heavily biased by non-Gaussianity of samples

From Van der Merwe et al.

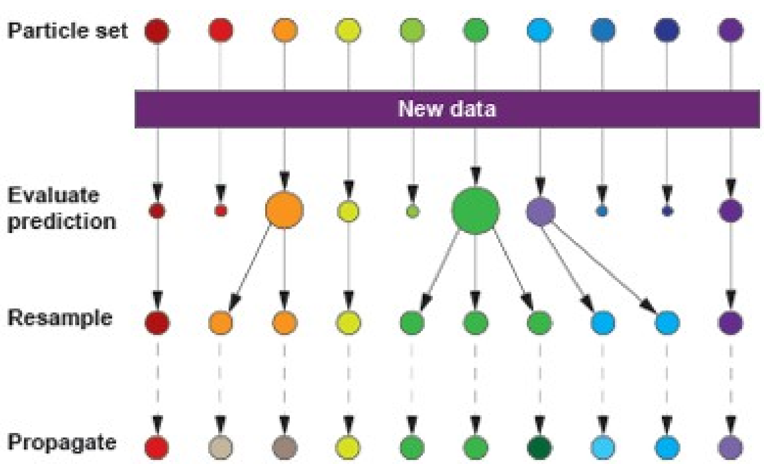
So we can propagate samples

So, rather than propagate the **sufficient statistics** (e.g. update the mean, variance) we can **sample from the posterior** over y[t-1] and then transform each sample to obtain a **sample set which describes the distribution of** y[t]

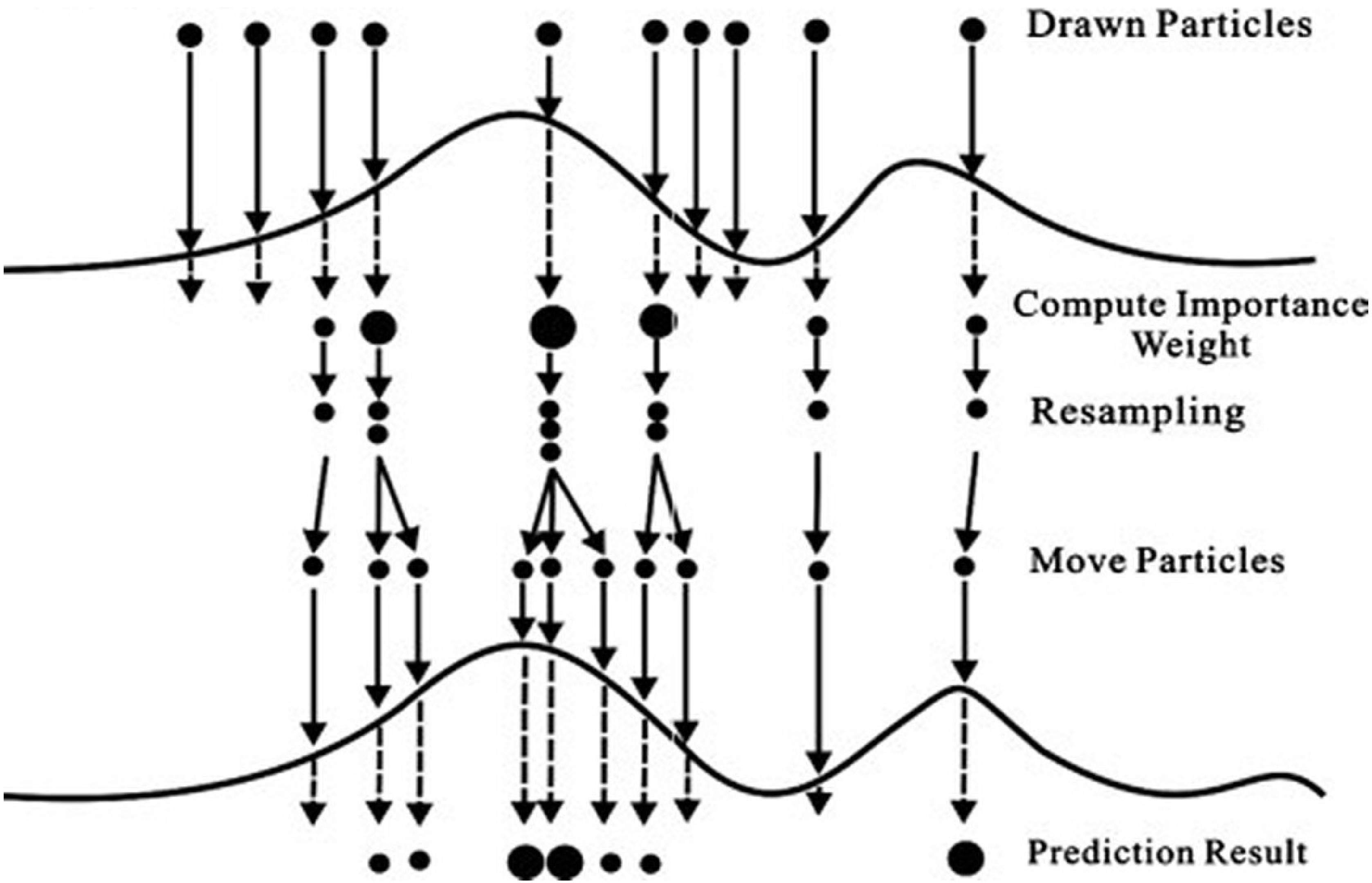
How do we sample?

- Use **importance sampling**

- Leads to seeing **particle filters** as **Successive Important Resampling** (SIR) filters



Importance Resampling



An example

Consider our system **state variable** to evolve under a transform like

$$\boldsymbol{a}_{t+1} = F\boldsymbol{a}_t + \boldsymbol{v}_t$$

Diffusion process

We can form the **prior** based on **past observations**

$$p(\boldsymbol{a}_t \,|\, X_{t-1}) = \int p(\boldsymbol{a}_t \,|\, \boldsymbol{a}_{t-1}) p(\boldsymbol{a}_{t-1} \,|\, X_{t-1}) \, d\boldsymbol{a}_{t-1}$$

We then observe the new datum $\boldsymbol{x}_t$

$$p(\boldsymbol{a}_t \,|\, X_t) = Z^{-1} p(\boldsymbol{x}_t \,|\, \boldsymbol{a}_t) p(\boldsymbol{a}_t \,|\, X_{t-1})$$

1) Draw samples from $p(\boldsymbol{a}_{t-1} \,|\, X_{t-1})$

2) Propagate through $\boldsymbol{a}_{t+1} = F\boldsymbol{a}_t + \boldsymbol{v}_t$

3) Get the importance weights $q_t^n = \dfrac{p(\boldsymbol{x}_t \,|\, \boldsymbol{a}_{t|t-1}^n)}{\sum_{k=1}^{N_p} p(\boldsymbol{x}_t \,|\, \boldsymbol{a}_{t|t-1}^k)}$ and thence $p(\boldsymbol{a}_t \,|\, X_t)$

4) iterate