Variational Inference, Gaussian Processes and non-linear state-space models

Carl Edward Rasmussen

Sensor Fusion 2018

July 11th, 2018

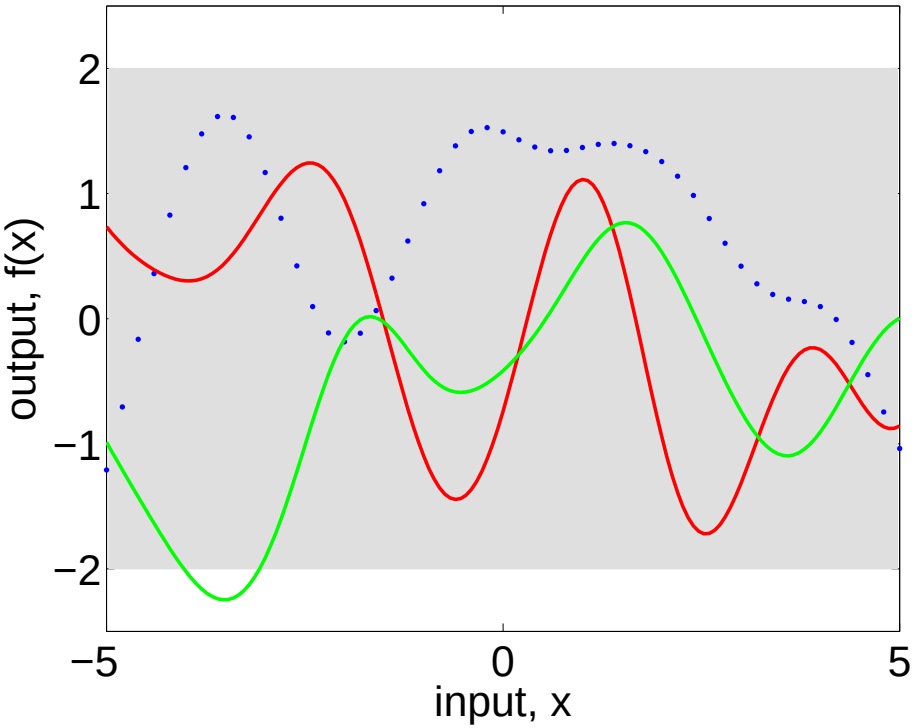
Properties of Gaussian Process models

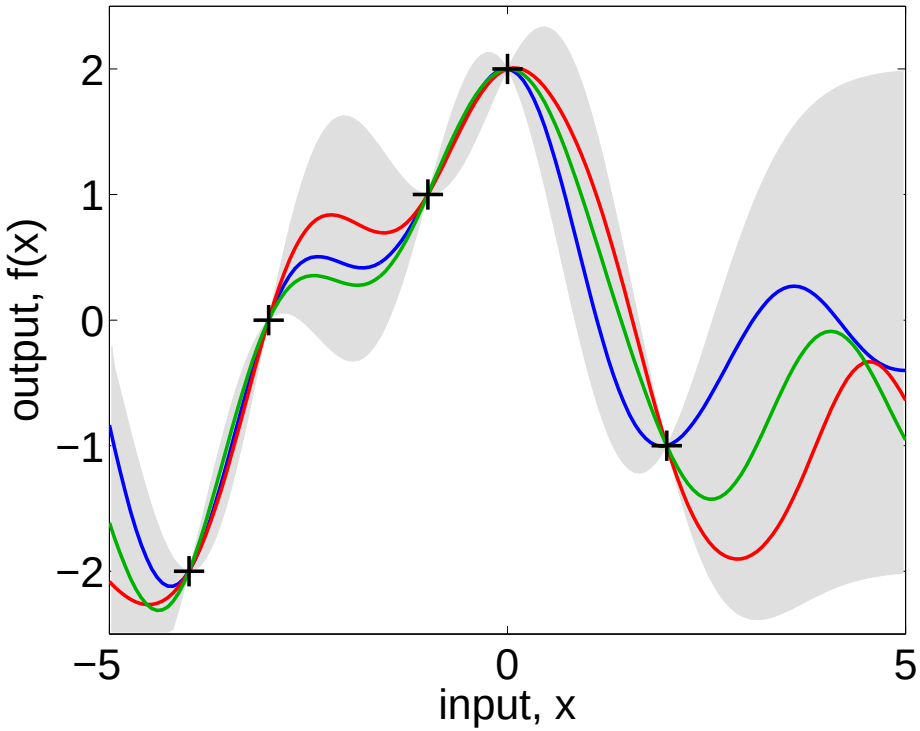
- allow analytic Bayesian inference
- make probabilistic (Gaussian) predictions
- marginal likelihood (for model selection) in closed form
- flexible, interpretable, non-parametric models

but

- simple, exact inference only available in vanilla cases
- exact inference scales badly with data set size

Gaussian Processes





Gaussian Processes

A Gaussian Process (GP) is a generalization of the Gaussian distribution to infinite dimensional objects.

An inifinitely long vector $\simeq$ a function.

Therefore, GPs specify distributions over functions

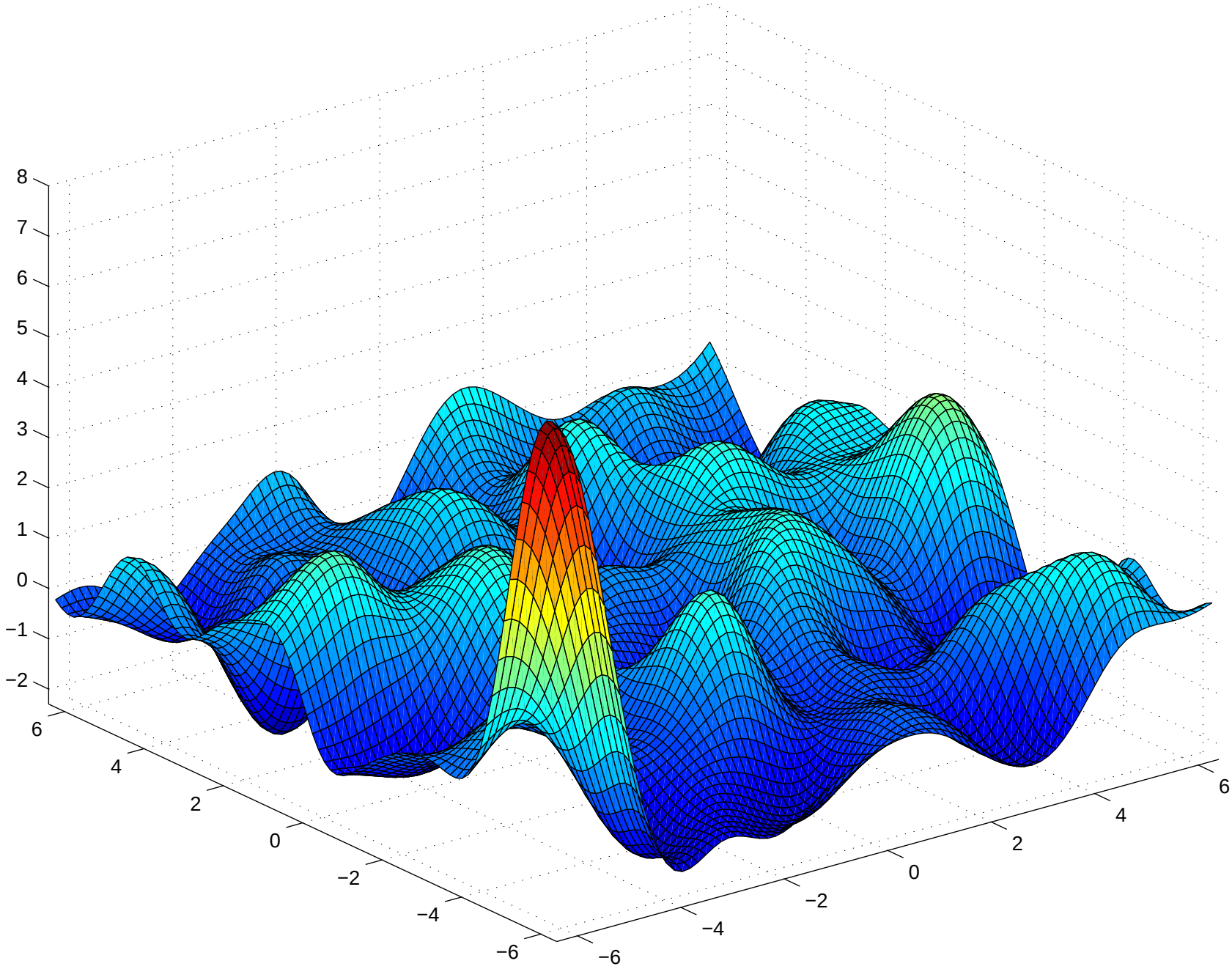
$$f(x) \sim \mathcal{GP}\big(m(x), k(x, x')\big).$$

A GP is fully specified by a mean function and a covariance function.

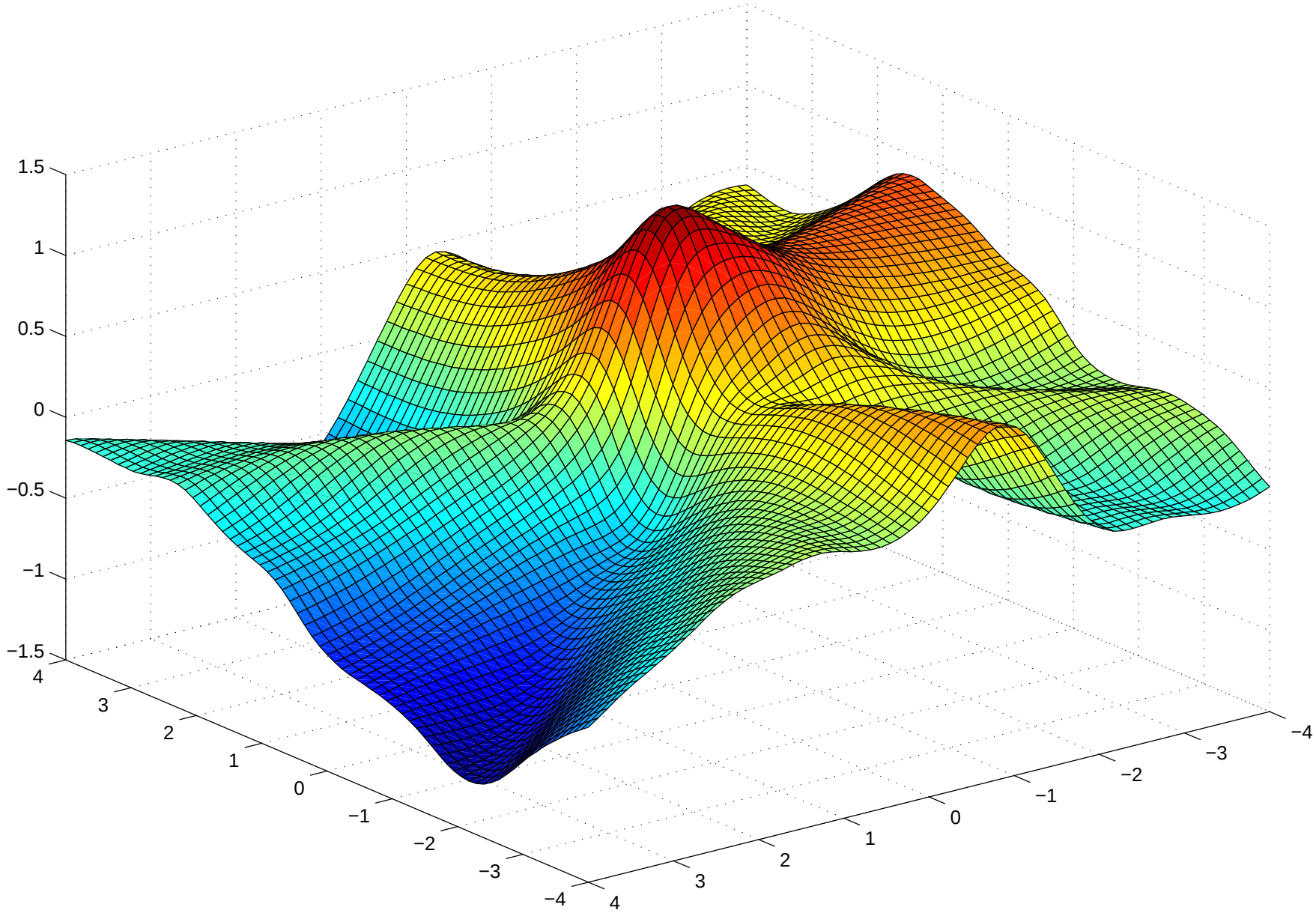
The properties of the function are controlled by the mean and covariance functions, and their hyperparameters θ.

Note: GPs are often used as *processes in time*. In this talk the index set into the process is more general.

Function drawn at random from a Gaussian Process with Gaussian covariance



Function drawn at random from a Neural Network covariance function



Important properties

Although functions are infinite dimensional objects, you can still do exact inference using finite memory and computation.

In probability theory this is called the marginalization property, in the kernel learning community it is called the *kernel trick*.

The prior is captured by a GP, after observing data, the posterior is also a GP with a new mean and covariance function.

The posterior predictive distribution at a test input x^* is

$$f(x^*)|\mathbf{y}, \theta \sim \mathcal{N}\big(k(x^*, x)[K(x, x) + \sigma^2 I]^{-1}\mathbf{y}, \\ k(x^*, x^*) - k(x^*, x)[K(x, x) + \sigma^2 I]^{-1}k(x, x^*)\big).$$

Model selection or hyperparameter learning

Bayesian model selection and hyperpameter optimization can be done using the marginal likelihood

$$\log p(\mathbf{y}|\theta) = \underbrace{-\tfrac{1}{2}\mathbf{y}^\top [K(x,x)+\sigma^2 I]^{-1}\mathbf{y}}_{\text{data fit}} \underbrace{-\tfrac{1}{2}\log|K(x,x)+\sigma^2 I|}_{\text{complexity term}} - \tfrac{N}{2}\log 2\pi.$$

Note: this is a non-parametric model, it can fit the data exactly if it wants to, but it doesn't, because we are marginalizing over functions (not optimizing). Occam's Razor is automatic.

Wouldn't it be nice if we could use GPs to solve complex non-linear modeling problems?

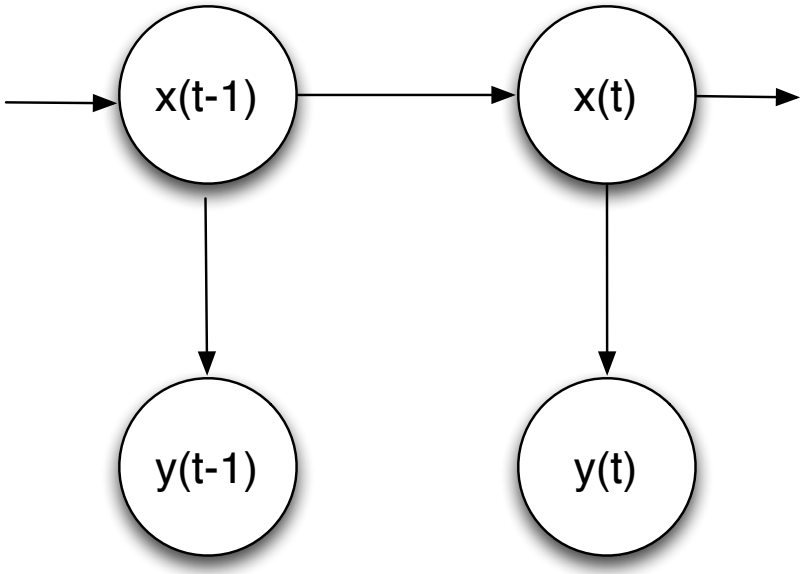
Issues

One issue is that inference takes $\mathcal{O}(N^3)$ computation and $\mathcal{O}(N^2)$ memory – doesn't scale to large problems.

Not clear how to handle stochastic inputs, as necessary in time-series models and learning in deep architectures.

The Gaussian posterior is only exact for Gaussian likelihood.

An inconvenient truth



Gaussian processes (GP) are an *extremely powerful framework* for learning and inference about non-linear functions.

It's the transition non-linearity which is essential; we will assume that the observation model is linear.

Irritating fact: Although desirable, it is not easy to apply GPs to latent variable time series.

The Evidence Lower BOund, ELBO

Start from Bayes rule

$$p(f|\mathbf{y},\theta) \;=\; \frac{p(\mathbf{y}|\mathbf{f})p(f|\theta)}{p(\mathbf{y}|\theta)} \quad\Leftrightarrow\quad p(\mathbf{y}|\theta) \;=\; \frac{p(\mathbf{y}|\mathbf{f})p(f|\theta)}{p(f|\mathbf{y},\theta)},$$

multiply and divide by an *arbitrary* distribution $q(f)$, and take logarithms

$$p(\mathbf{y}|\theta) \;=\; \frac{p(\mathbf{y}|\mathbf{f})p(f|\theta)}{p(f|\mathbf{y},\theta)}\frac{q(f)}{q(f)} \quad\Leftrightarrow\quad \log p(\mathbf{y}|\theta) \;=\; \log\frac{p(\mathbf{y}|\mathbf{f})p(f|\theta)}{q(f)} + \log\frac{q(f)}{p(f|\mathbf{y},\theta)},$$

average both sides wrt $q(f)$

$$\underbrace{\log p(\mathbf{y}|\theta)}_{\text{marginal likelihood}} \;=\; \underbrace{\int q(f)\log\frac{p(\mathbf{y}|\mathbf{f})p(f|\theta)}{q(f)}df}_{\text{Evidence Lower BOund, ELBO}} + \underbrace{\int q(f)\log\frac{q(f)}{p(f|\mathbf{y},\theta)}df}_{\mathcal{KL}\left(q(f)||p(f|\mathbf{y},\theta)\right)}.$$

Variational Inference in Gaussian Processes

Algorithm: maximize ELBO wrt $q(f)$ within some restricted class; key assumption

$$\boxed{q(f) \;=\; p(f_{\neq \mathbf{u}}|\mathbf{u},\theta)q(\mathbf{u}),}$$

where f is split into two disjont sets, the *inducing targets* $\mathbf{u}$, and the rest $f_{\neq \mathbf{u}}$.

$$\begin{aligned}
\mathcal{F}(\theta) &= \int q(f)\log\frac{p(\mathbf{y}|\mathbf{f})p(f|\theta)}{q(f)}df \\
&= \int p(f_{\neq \mathbf{u}}|\mathbf{u})q(\mathbf{u})\log\frac{p(\mathbf{y}|\mathbf{f})\cancel{p(f_{\neq \mathbf{u}}|\mathbf{u},\theta)}p(\mathbf{u}|\theta)}{\cancel{p(f_{\neq \mathbf{u}}|\mathbf{u},\theta)}q(\mathbf{u})}df_{\neq \mathbf{u}}d\mathbf{u} \\
&= \sum_{n=1}^{N}\int p(f_n|\mathbf{u},\theta)q(\mathbf{u})\log p(y_n|f_n)d\mathbf{u}df_n - \mathcal{KL}(q(\mathbf{u})||p(\mathbf{u}|\theta)),
\end{aligned}$$

which can be evaluated in closed form.

Variantional Inference in Gaussian Processes, cont

The lower bound can be evaluated in closed form.

The lower bound can be optimized wrt $q(\mathbf{u})$ by free form optimisation.

The optimal $q^*(\mathbf{u})$ turns out to be Gaussian.

Plugging the optimal Gaussian back into the bound, we get

$$\mathcal{F}(\theta) \;=\; -\tfrac{1}{2}\mathbf{y}^\top [Q_{\mathbf{ff}} + \sigma^2 I]^{-1}\mathbf{y} - \tfrac{1}{2}\log|Q_{\mathbf{ff}} + \sigma^2 I| - \tfrac{1}{2\sigma^2}\mathrm{tr}(K_{\mathbf{ff}} - Q_{\mathbf{ff}}) - \tfrac{N}{2}\log(2\pi),$$

where $Q_{\mathbf{ff}} = K_{\mathbf{fu}} K_{\mathbf{uu}}^{-1} K_{\mathbf{uf}}$. The structure (low rank plus diagonal), allows evaluation in $\mathcal{O}(M^2 N)$, where M is the number of inducing targets.

The bound can be optimized wrt hyperparameters θ and inducing inputs $\mathbf{z}$.

Generative Model

Gaussian Process State Space Model

$$
\begin{aligned}
f(\mathbf{x}|\theta_x) &\sim \mathcal{GP}\big(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\big), && (1)\\
\mathbf{x}_t|\mathbf{f}_t &\sim \mathcal{N}(\mathbf{x}_t|\mathbf{f}_t, Q), \qquad Q \text{ diagonal}, && (2)\\
\mathbf{y}_t|\mathbf{x}_t &\sim \mathcal{N}(\mathbf{y}_t|C\mathbf{x}_t, R), && (3)
\end{aligned}
$$

with hyperparameters $\theta = (\theta_x, Q, C, R)$.

The joint probability is the prior times the likelihood

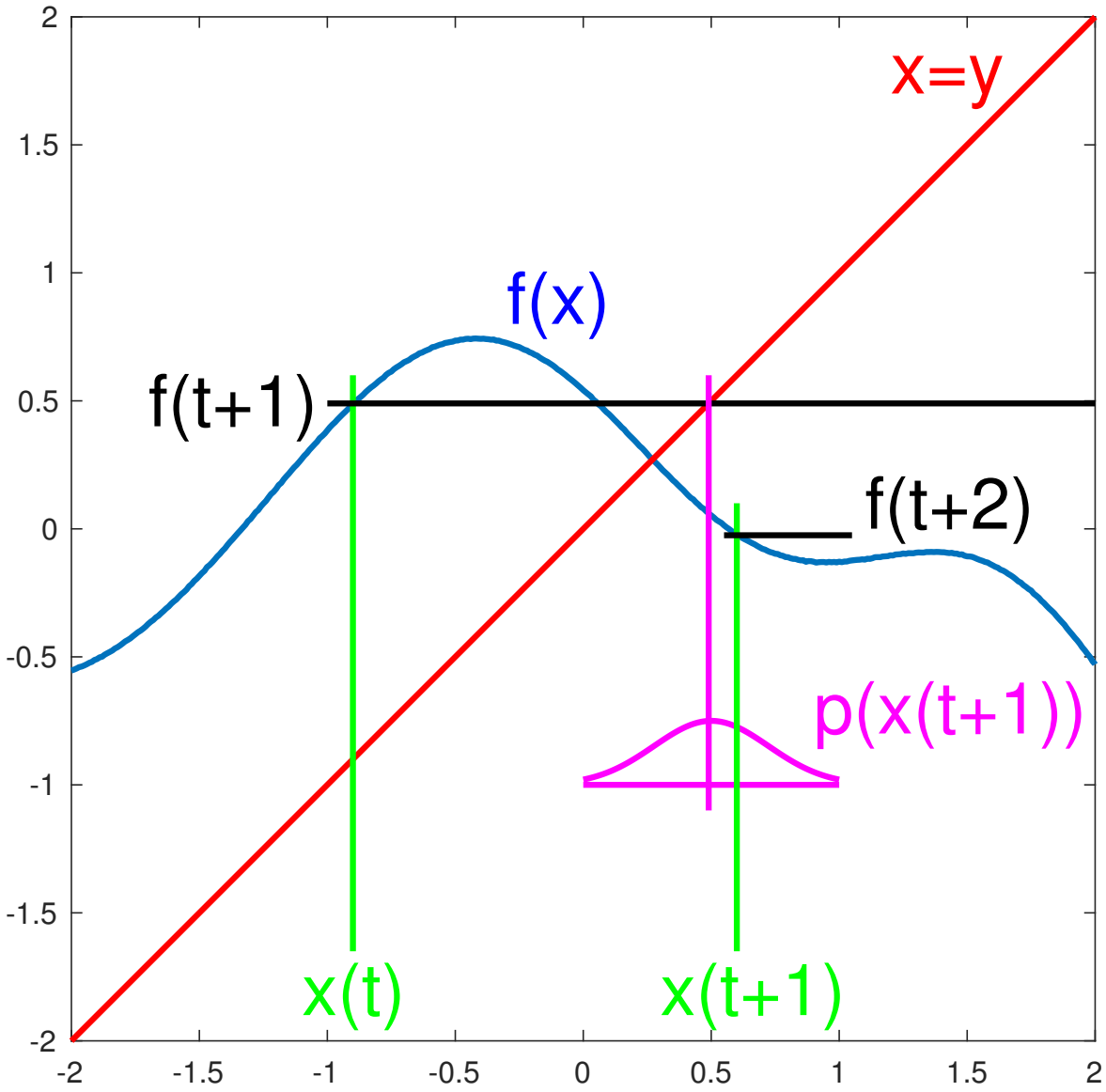
$$
p(\mathbf{y}, \mathbf{x}, \mathbf{f}|\theta) = p(\mathbf{x}_0) \prod_{t=1}^{T} p(f_t|\mathbf{f}_{1:t-1}, \mathbf{x}_{0:t-1}, \theta_x) p(x_t|f_t, Q) p(y_t|x_t, C, R). \qquad (4)
$$

Note that the joint distribution of the $\mathbf{f}$ values isn't even Gaussian! The marginal likelihood is

$$
p(\mathbf{y}|\theta) = \int p(\mathbf{y}, \mathbf{x}, \mathbf{f}|\theta) d\mathbf{f} d\mathbf{x}. \qquad (5)
$$

That's *really* awful! But we need it to train the model.

Picture of part of generative process



Let's lower bound the marginal likelihood

$$\begin{aligned}
\log p(\mathbf{y}|\theta) &\geqslant \int q(\mathbf{x},\mathbf{f}) \log \frac{p(\mathbf{x}_0)\prod_{t=1}^{T} p(f_t|\mathbf{f}_{1:t-1},\mathbf{x}_{0:t-1})p(x_t|f_t)p(y_t|x_t)}{q(\mathbf{x},\mathbf{f})} d\mathbf{x}d\mathbf{f} \\
&= \int q(\mathbf{x},\mathbf{f}) \log \frac{p(\mathbf{y}|\theta)p(\mathbf{x},\mathbf{f}|\mathbf{y},\theta)}{q(\mathbf{x},\mathbf{f})} d\mathbf{x}d\mathbf{f} \qquad (6) \\
&= \log p(\mathbf{y}|\theta) - \mathcal{KL}(q(\mathbf{x},\mathbf{f})||p(\mathbf{x},\mathbf{f}|\mathbf{y},\theta)).
\end{aligned}$$

for any distribution $q(\mathbf{x},\mathbf{f})$, by Jensen's inequality.

Let's chose the $q(\mathbf{x},\mathbf{f})$ distribution within some restricted family to maximize the lower bound or equivalently minimize the $\mathcal{KL}$ divergence.

This is still nasty because of the annoying prior, unless we chose:

$$q(\mathbf{x},\mathbf{f}) = q(\mathbf{x})\prod_{t=1}^{T} p(f_t|\mathbf{f}_{1:t-1},\mathbf{x}_{0:t-1}).$$

This choice makes the lower bound simple but terribly loose! Why? Because the approximating distribution over the $\mathbf{f}$'s doesn't depend on the observations $\mathbf{y}$.

Augment GP with inducing variables

Augment the model with an additional set of input output pairs $\{\mathbf{z}_i, u_i | i = 1, \ldots M\}$

Joint distribution:

$$p(\mathbf{y}, \mathbf{x}, \mathbf{f}, \mathbf{u}|\mathbf{z}) \;=\; p(\mathbf{x}, \mathbf{f}|\mathbf{u}) p(\mathbf{u}|\mathbf{z}) \prod_{t=1}^{T} p(y_t|x_t). \tag{7}$$

Consistency (or the marginalization property) of GPs ensures that it is straight forward to augment with extra variables.

This step seemingly makes our problem *worse*, because we have more latent variables.

Lower bound revisited

Lower bound on marginal likelihood

$$\log p(\mathbf{y}|\theta) \geqslant \int q(\mathbf{x},\mathbf{f},\mathbf{u}) \log \frac{p(\mathbf{u})p(\mathbf{x}_0)\prod_{t=1}^{T} p(f_t|\mathbf{f}_{1:t-1},\mathbf{x}_{0:t-1},\mathbf{u})p(x_t|f_t)p(y_t|x_t)}{q(\mathbf{x},\mathbf{f},\mathbf{u})} d\mathbf{x}d\mathbf{f}d\mathbf{u}, \quad (8)$$

for any distribution $q(\mathbf{x},\mathbf{f},\mathbf{u})$, by Jensen's inequality.

Now chose

$$q(\mathbf{x},\mathbf{f},\mathbf{u}) = q(\mathbf{u})q(\mathbf{x})\prod_{t=1}^{T} p(f_t|\mathbf{f}_{1:t-1},\mathbf{x}_{0:t-1},\mathbf{u}). \quad (9)$$

This choice conveniently makes the troublesome $p(f_t|\mathbf{f}_{1:t-1},\mathbf{x}_{0:t-1},\mathbf{u})$ term cancel. But the $\mathbf{u}$ values can be chosen (via $q(\mathbf{u})$) to reflect the observations, so the bound may be tight(er).

The lower bound is $\mathcal{L}(\mathbf{y}|q(\mathbf{x}),q(\mathbf{u}),q(\mathbf{f}|\mathbf{x},\mathbf{u}),\theta)$.

What do we get for all our troubles?

- For certain choices of covariance function, $\mathbf{f}$ can be integrated out

$$\mathcal{L}(\mathbf{y}|q(\mathbf{x}), q(\mathbf{u}), \theta) = \int \mathcal{L}(\mathbf{y}|q(\mathbf{x}), q(\mathbf{u}), q(\mathbf{f}|\mathbf{x}, \mathbf{u}), \theta) d\mathbf{f} \tag{10}$$

- The optimal $q(\mathbf{u})$ is found by calculus is variations and turns out to be Gaussian, and can be maxed out

$$\mathcal{L}(\mathbf{y}|q(\mathbf{x}), \theta) = \max_{q(\mathbf{u})} \mathcal{L}(\mathbf{y}|q(\mathbf{x}), q(\mathbf{u}), \theta) \tag{11}$$

- The optimal $q(\mathbf{x})$ has Markovian structure; making a further Gaussian assumption, the lower bound can be evaluated analytically, as a function of its parameters μ_t, $\Sigma_{t,t}$ and $\Sigma_{t-1,t}$.

Algorithm: optimize the lower bound $\mathcal{L}(\mathbf{y}|q(\mathbf{x}), \theta)$ wrt the parameters of the Gaussian $q(\mathbf{x})$ and the remaining parameters θ (we need derivatives for the optimisation).

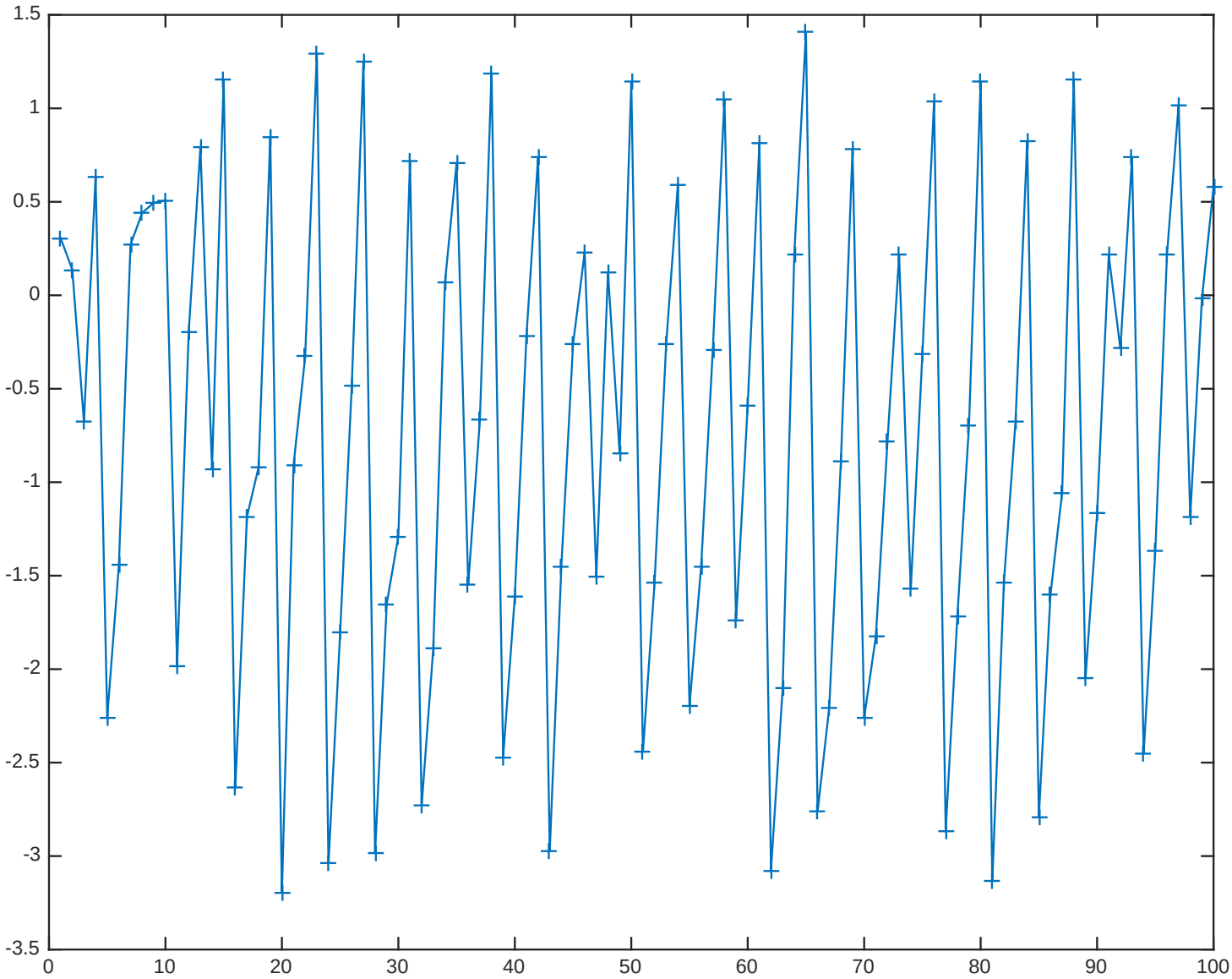
Sparse approximations are builtin

The computational cost is dominated by the GP. But, the effective 'training set size' for the GP is given by M the number of inducing variables.

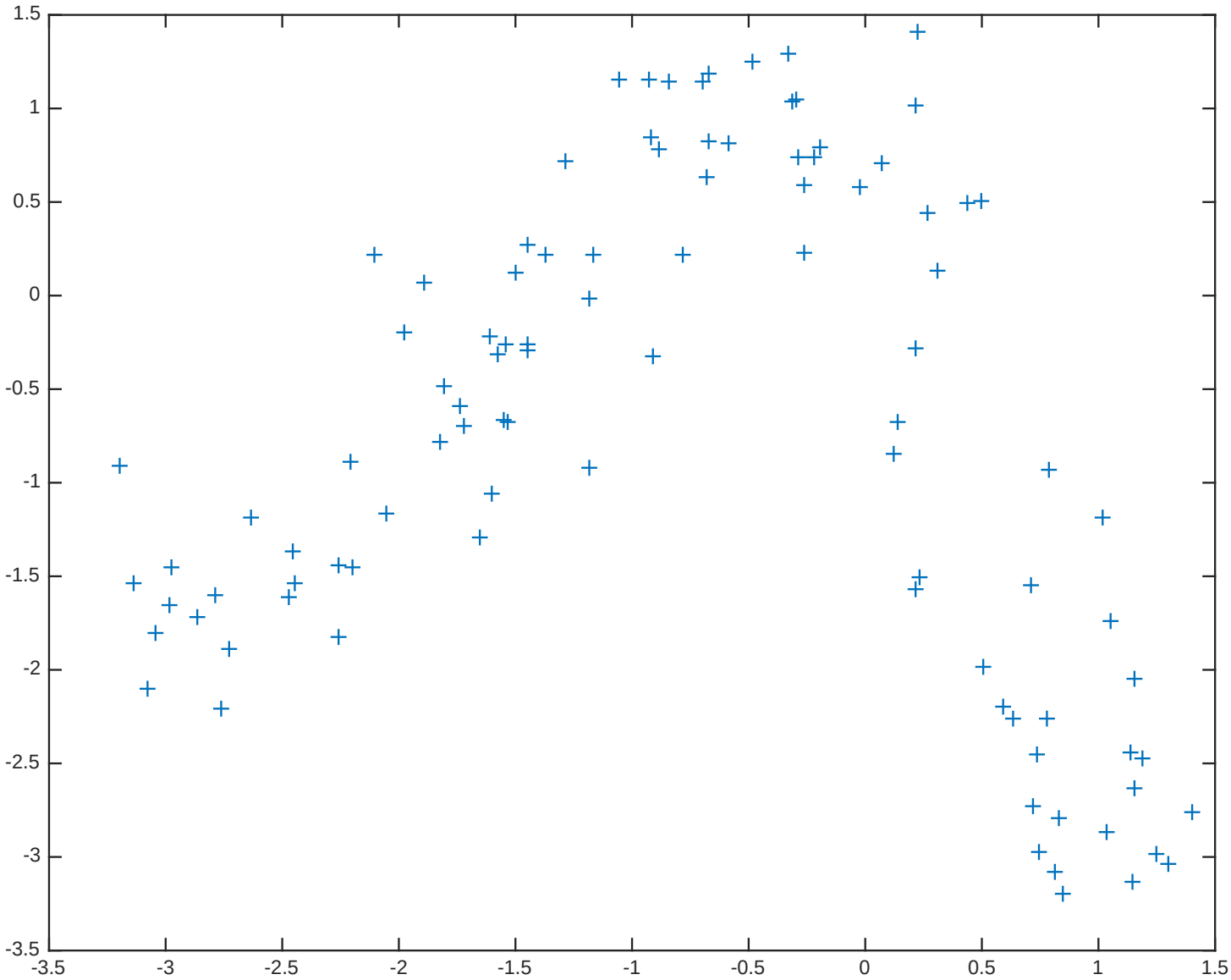
Therefore, we can chose M to trade off accuracy and computational demand.

The computational cost is linear in the length of the time-series.

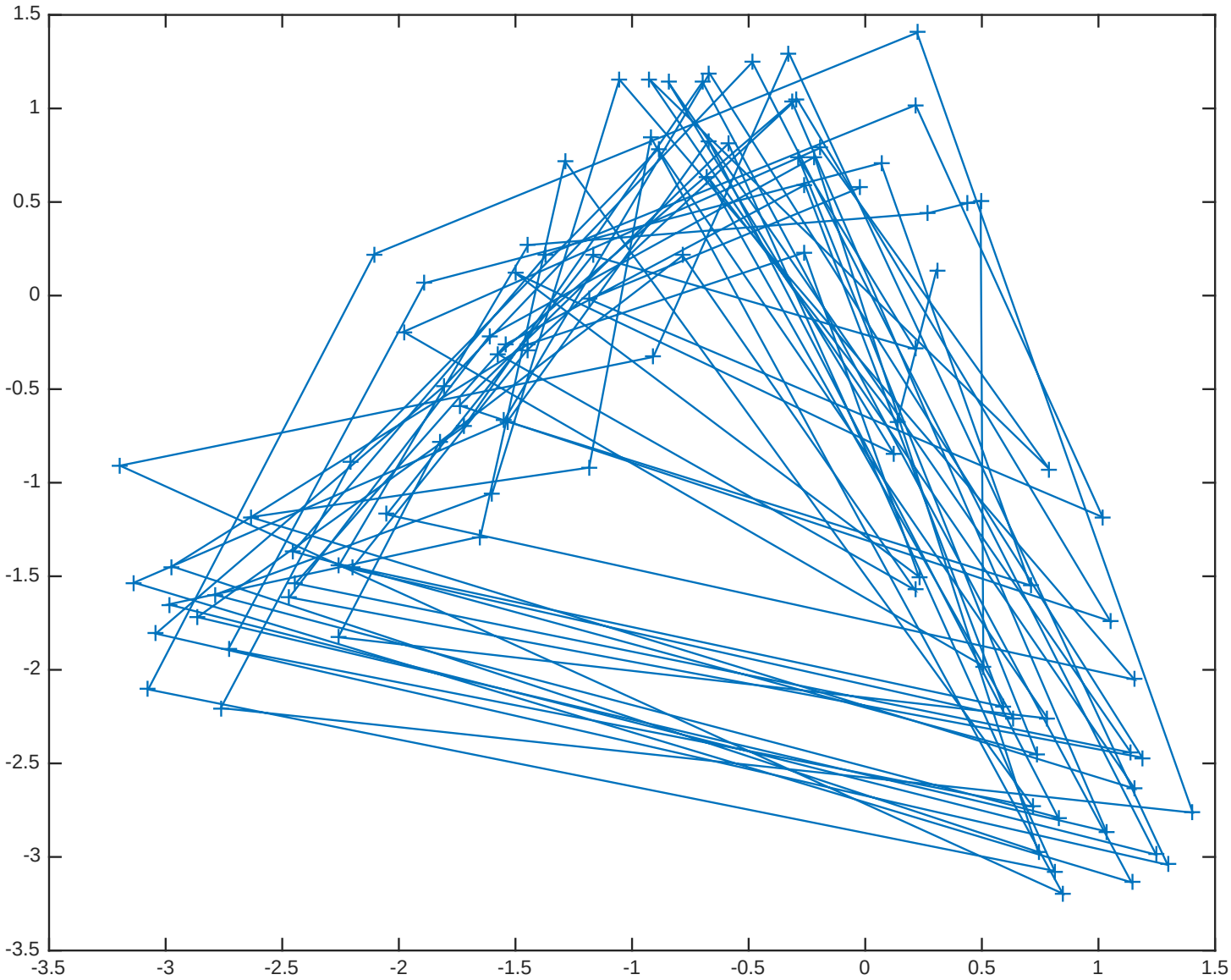
Data from non-linear Dynamical System



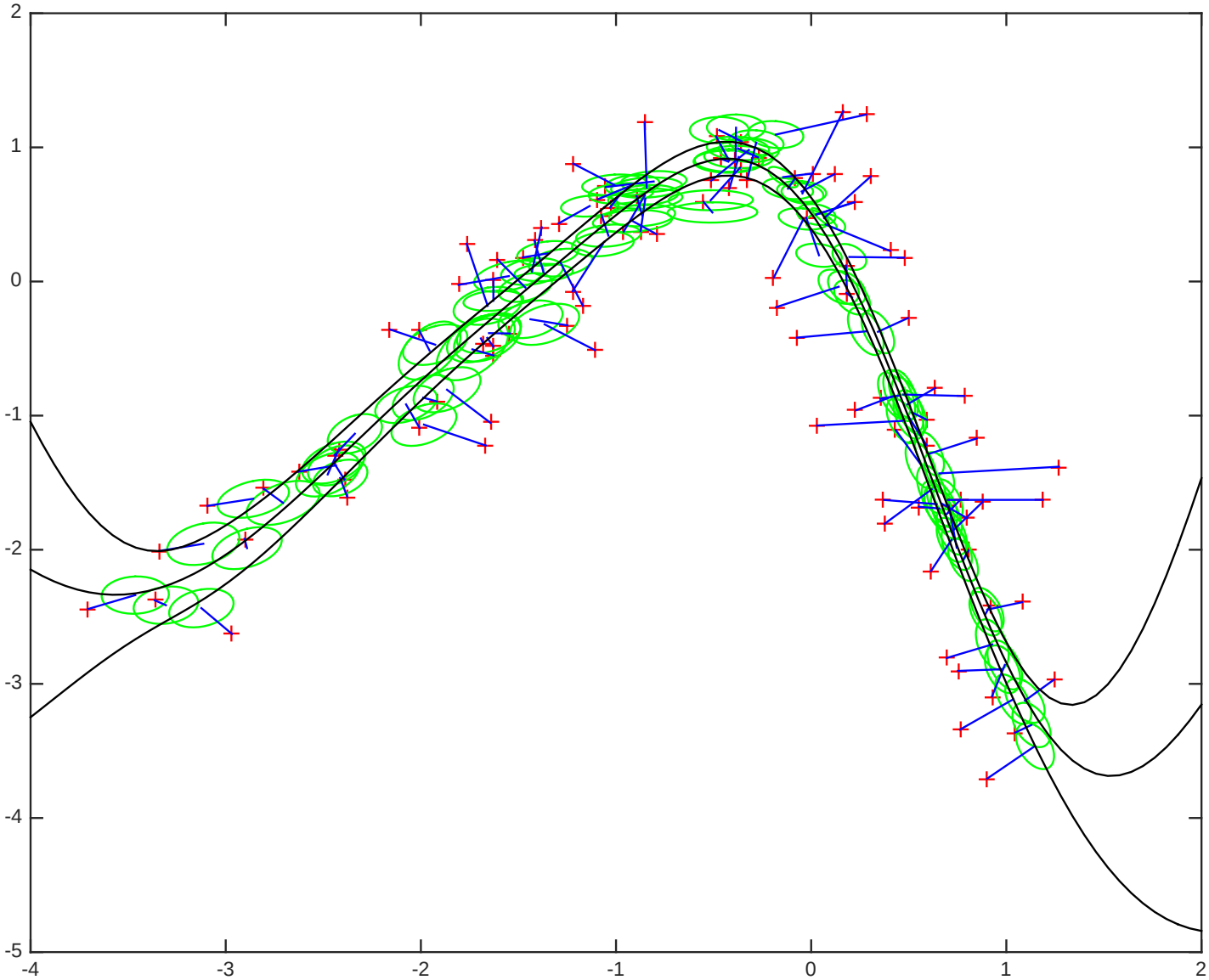
State space representation: f(t) as a function of f(t-1)



With explicit transitions



Data from non-linear Dynamical System



Parameterisation

Let's use the method on a sequence of length T, with D dimensional observations and a D dimensional latent state, ie we have TD observations.

The lower bound is to be optimized wrt all the parameters

- the pairwise marginal Gaussian distribution $q(\mathbf{x})$. This contains TD parameters for the mean and $2TD^2$ for the covariances.
- the inducing inputs z. For each of the D GPs there are MD inducing inputs, ie a total of MD^2.
- parameters of the observation model C, there are D^2 of these
- noise parameters, $2D$.
- GP hyperparameters, $\sim D^2$.

for a grand total of roughly $(2T + M)D^2$.

Example: cart and inverted pendulum, $D = 4$ and 10 timeseries each of length 100, so $T = 1000$ and $M = 50$. So we have 4000 observations and 36832 free parameters. This large number of parameters may be inconvenient but it doesn't lead to overfitting!

Implementation

Careful implementation is necessary

- $q(\mathbf{x})$ is assumed Markovian. Thus the *precision* matrix is block tridiagonal. The covariance is full, don't write it out, it's huge!
- $q(\mathbf{x})$ has to be parameterised carefully to allow all pos def matrices, but without writing out the covariances.
- initialization of parameters is important
  - most of the free parameters are in the covariance of $q(\mathbf{x})$. Initially train with *shared* covariances across time.
  - then continue training with free covariances.

Conclusions

- Principled, approximate inference in flexible non-linear non-parametric models for time series are possible and practical
- Allows to integrate over dynamics
- Automatically discover dimensionality of the hidden space – it is not the case that more latent dimensions lead to higher marginal likelihood

Some other interesting properties include

- Handles missing variables
- Multivariate latent variables and observations straight forward
- flexible non-parametric dynamics model
- Occam's Razor automatically at work, no overfitting
- Framework includes computational control by limiting M