Gaussian Processes as a Statistical Method

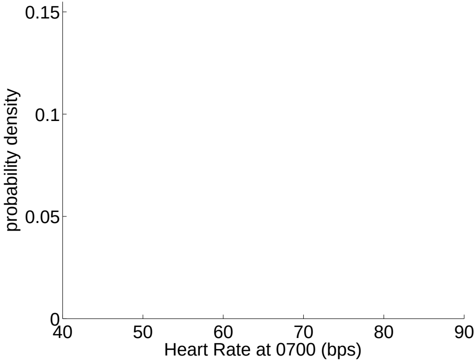
John P. Cunningham
Columbia University
Department of Statistics

February 11, 2014

Outline

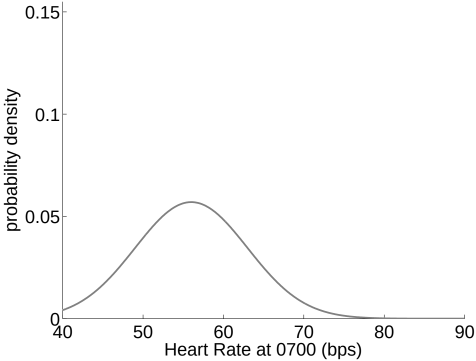
What is a Gaussian (for machine learning)?

- A handy tool for Bayesian inference on real valued variables:



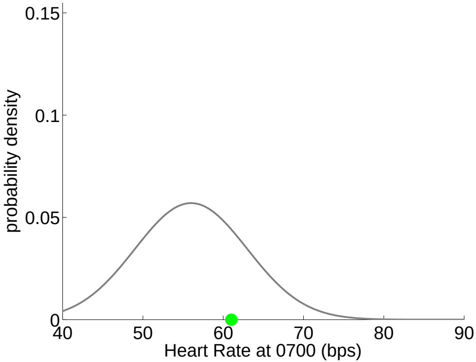
What is a Gaussian (for machine learning)?

- A handy tool for Bayesian inference on real valued variables:



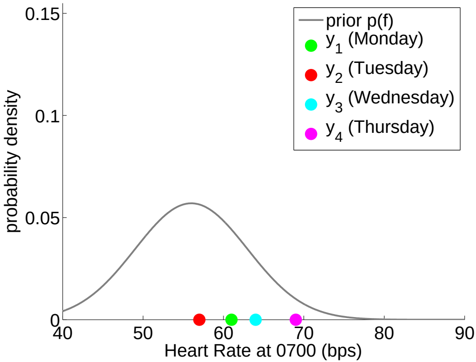
What is a Gaussian (for machine learning)?

- A handy tool for Bayesian inference on real valued variables:



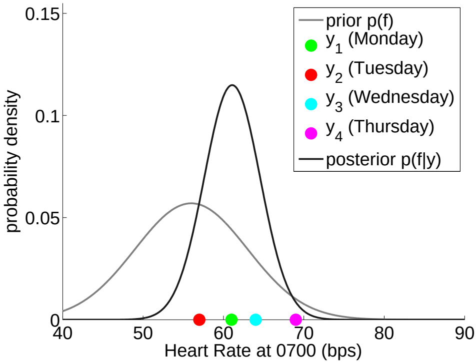
What is a Gaussian (for machine learning)?

- A handy tool for Bayesian inference on real valued variables:

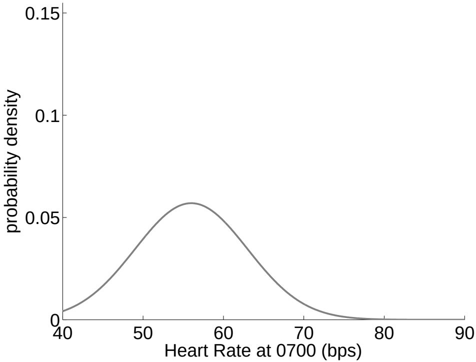


What is a Gaussian (for machine learning)?

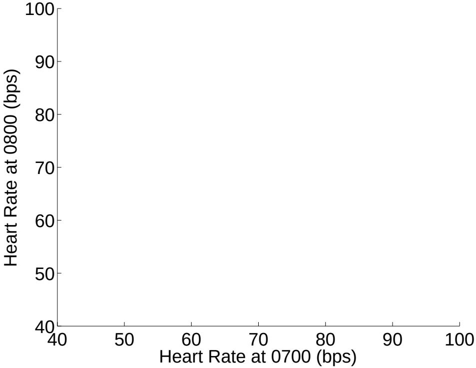
- A handy tool for Bayesian inference on real valued variables:



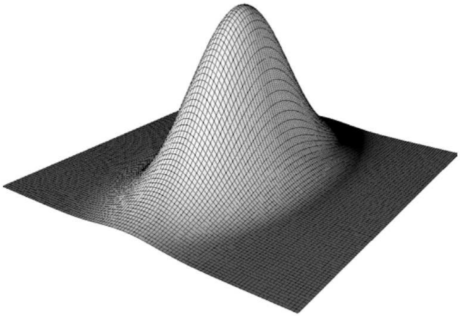
From univariate to multivariate Gaussians:



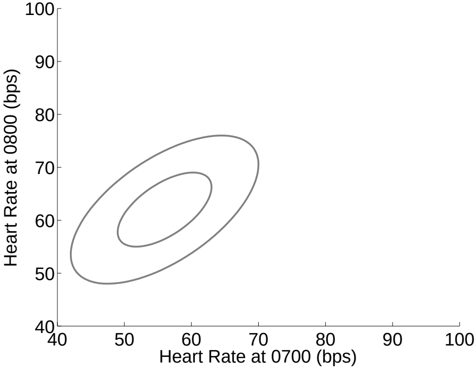
From univariate to multivariate Gaussians:



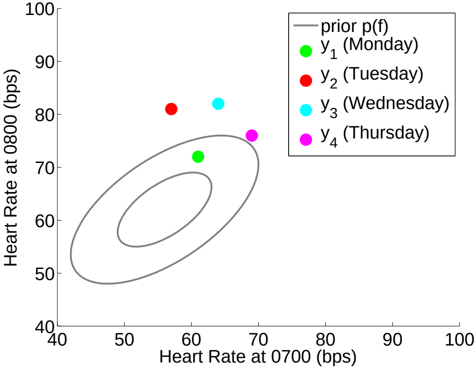
From univariate to multivariate Gaussians:



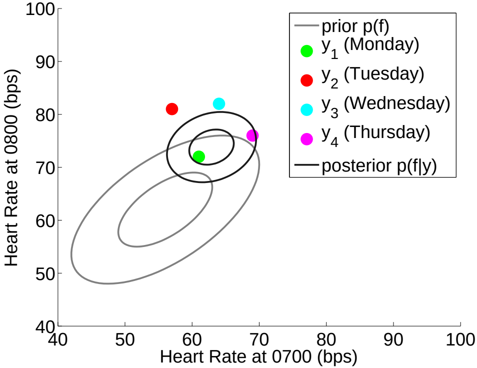
From univariate to multivariate Gaussians:



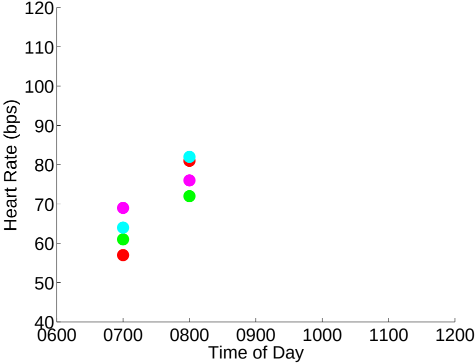
From univariate to multivariate Gaussians:



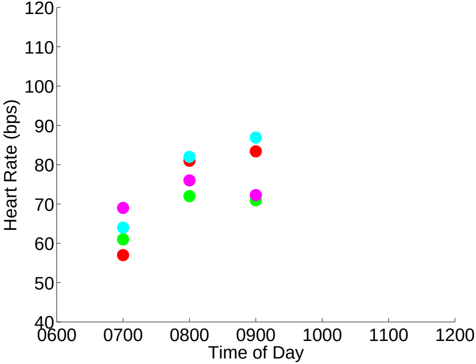
From univariate to multivariate Gaussians:



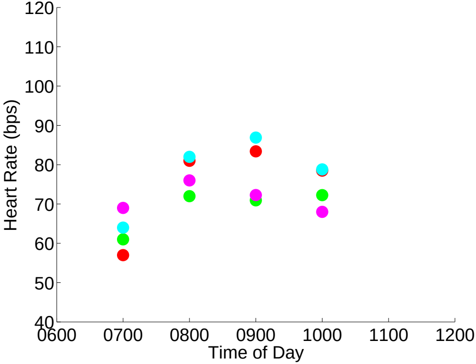
From multivariate Gaussians to Gaussian Processes:



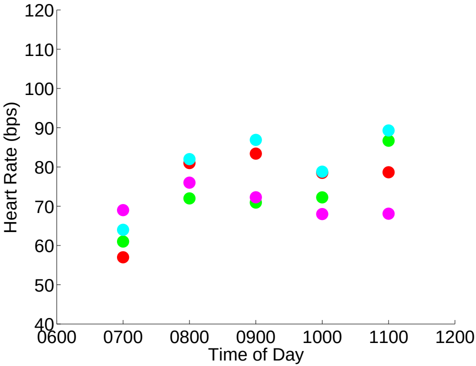
From multivariate Gaussians to Gaussian Processes:



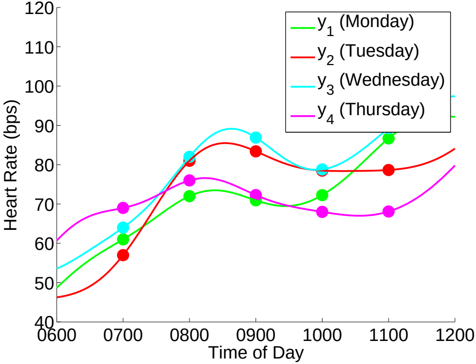
From multivariate Gaussians to Gaussian Processes:



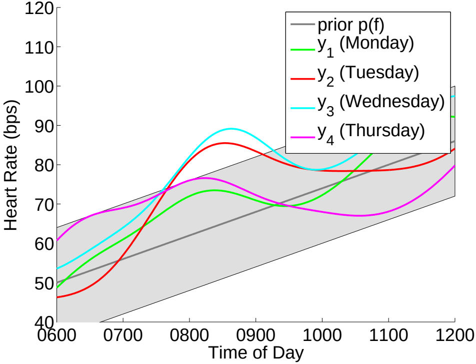
From multivariate Gaussians to Gaussian Processes:



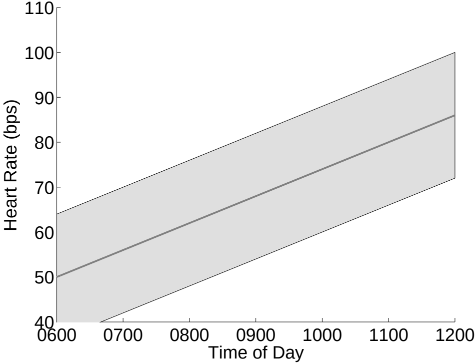
From multivariate Gaussians to Gaussian Processes:



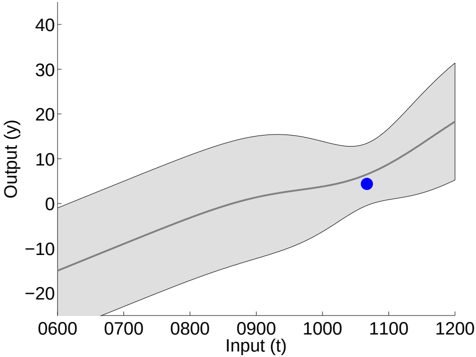
Our representation of a GP distribution:



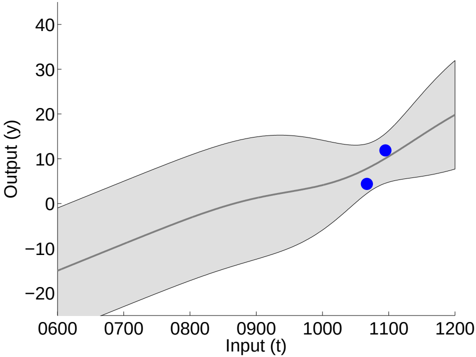
We can take measurements less rigidly:



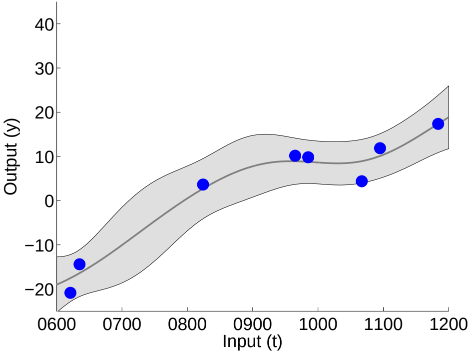
We can take measurements less rigidly:



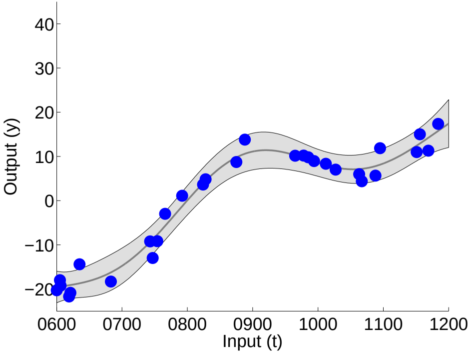
Updating the posterior:



Updating the posterior:



Updating the posterior:

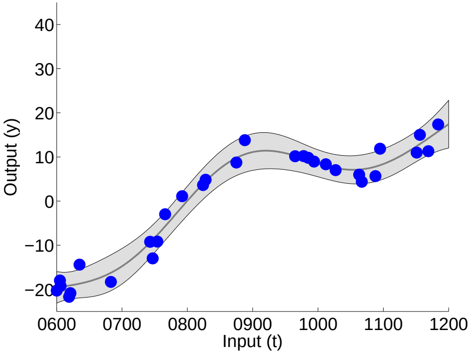


An intuitive summary

- Univariate Gaussians: distributions over real valued variables
- Multivariate Gaussians: {pairs, triplets, ... } of real valued vars
- Gaussian Processes: functions of (infinite numbers of) real valued variables $\rightarrow$ regression.

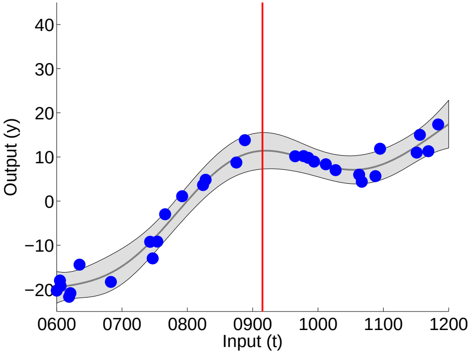
Regression: a few reminders

- denoising/smoothing



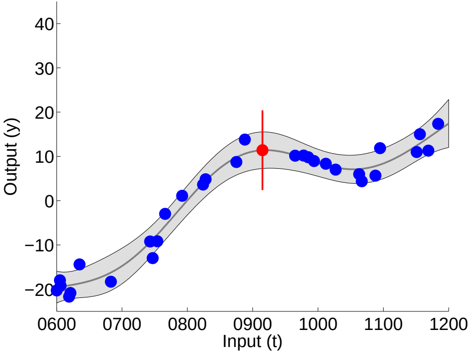
Regression: a few reminders

- denoising/smoothing
- prediction/forecasting



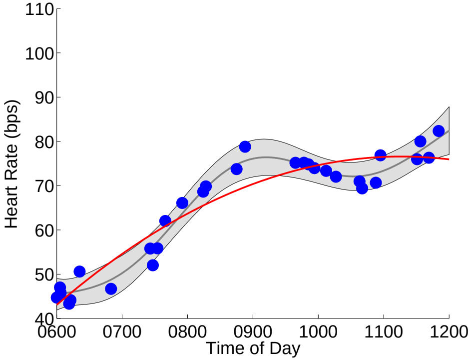
Regression: a few reminders

- denoising/smoothing
- prediction/forecasting



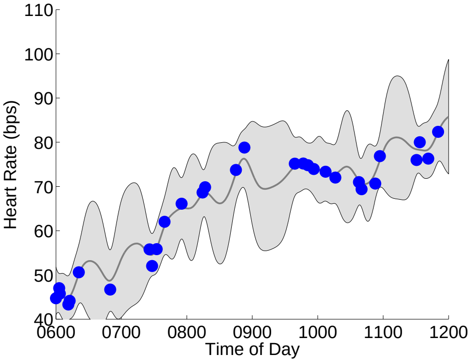
Regression: a few reminders

- denoising/smoothing
- prediction/forecasting
- dangers of parametric models



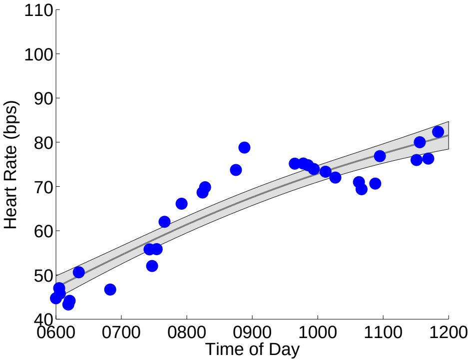
Regression: a few reminders

- denoising/smoothing
- prediction/forecasting
- dangers of parametric models
- dangers of overfitting/underfitting



Regression: a few reminders

- denoising/smoothing
- prediction/forecasting
- dangers of parametric models
- dangers of overfitting/underfitting



Outline

Review: multivariate Gaussian

- $f \in \mathbb{R}^n$ is normally distributed $\Leftrightarrow$

$$p(f) = (2\pi)^{-\frac{n}{2}} |K|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(f-m)^T K^{-1} (f-m)\right\}$$

- mean $m \in \mathbb{R}^n$ and (pd) covariance $K \in \mathbb{R}^{n \times n}$
- shorthand: $f \sim \mathcal{N}(m, K)$

Definition: Gaussian Process

- Loosely, a multivariate Gaussian of uncountably infinite length... really long vector $\approx$ function
- f is a Gaussian process if $f(t) = [f(t_1), ..., f(t_n)]'$ has a multivariate normal distribution for all $t = [t_1, ..., t_n]'$:

$$f(t) \sim \mathcal{N}(m(t), K(t, t))$$

- ($t \in \mathbb{R}$ as regression in time, but domain can be $x \in \mathbb{R}^D$)
- What are $m(t), K(t, t)$?

Definition: Gaussian Process

Mean function $m(t)$:

- any function $m : \mathbb{R} \to \mathbb{R}$ (or $m : \mathbb{R}^D \to \mathbb{R}$)
- very often $m(t) = 0 \ \forall \ t$ (mean subtract your data)

Kernel (covariance) function:

- any valid Mercer kernel $k : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$
- Mercer's theorem: every matrix $K(t,t) = \{k(t_i, t_j)\}_{i,j=1\ldots n}$ is a positive semidefinite (covariance) matrix $\forall t$:

$$v^T K(t,t) v = \sum_{i=1}^{n} \sum_{j=1}^{n} K_{ij} v_i v_j = \sum_{i=1}^{n} \sum_{j=1}^{n} k(t_i, t_j) v_i v_j \geq 0$$

Definition: Gaussian Process

GP is fully defined by:

- mean function $m(\cdot)$ and kernel (covariance) function $k(\cdot, \cdot)$
- requirement that every finite subset of the domain t has a multivariate normal $f(t) \sim \mathcal{N}(m(t), K(t,t))$

Notes

- that this should exist is not trivial!
- most interesting properties are inherited
- Kernel function...

Kernel Function

Example kernel (squared exponential or SE):

$$k(t_i, t_j) = \sigma_f^2 \exp\left\{ -\frac{1}{2\ell^2}(t_i - t_j)^2 \right\}$$

From kernel to covariance matrix

- Choose some *hyperparameters*: $\sigma_f = 7$, $\ell = 100$

$$t = \begin{bmatrix} 0700 \\ 0800 \\ 1029 \end{bmatrix} \qquad K(t,t) = \{k(t_i, t_j)\}_{i,j} = \begin{bmatrix} 49.0 & 29.7 & 00.2 \\ 29.7 & 49.0 & 03.6 \\ 00.2 & 03.6 & 49.0 \end{bmatrix}$$

Kernel Function

Example kernel (squared exponential or SE):

$$k(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$$

From kernel to covariance matrix

- Choose some *hyperparameters*: $\sigma_f = 7$, $\ell = 500$

$$t = \begin{bmatrix} 0700 \\ 0800 \\ 1029 \end{bmatrix} \qquad K(t,t) = \{k(t_i, t_j)\}_{i,j} = \begin{bmatrix} 49.0 & 48.0 & 39.5 \\ 48.0 & 49.0 & 44.1 \\ 39.5 & 44.1 & 49.0 \end{bmatrix}$$

Kernel Function

Example kernel (squared exponential or SE):

$$k(t_i, t_j) = \sigma_f^2 \exp\left\{ -\frac{1}{2\ell^2}(t_i - t_j)^2 \right\}$$

From kernel to covariance matrix

- Choose some *hyperparameters*: $\sigma_f = 7$, $\ell = 50$

$$t = \begin{bmatrix} 0700 \\ 0800 \\ 1029 \end{bmatrix} \qquad K(t,t) = \{k(t_i,t_j)\}_{i,j} = \begin{bmatrix} 49.0 & 06.6 & 00.0 \\ 06.6 & 49.0 & 00.0 \\ 00.0 & 00.0 & 49.0 \end{bmatrix}$$

Kernel Function

Example kernel (squared exponential or SE):

$$k(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$$

From kernel to covariance matrix

- Choose some *hyperparameters*: $\sigma_f = 14$, $\ell = 50$

$$t = \begin{bmatrix} 0700 \\ 0800 \\ 1029 \end{bmatrix} \qquad K(t,t) = \{k(t_i, t_j)\}_{i,j} = \begin{bmatrix} 196 & 26.5 & 00.0 \\ 26.5 & 196 & 0.01 \\ 00.0 & 0.01 & 196 \end{bmatrix}$$

Kernels: looking ahead at computation

Example kernel (squared exponential or SE):

$$k(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$$

What happens if our time points are equally spaced?

- Choose some *hyperparameters*: $\sigma_f = 7$, $\ell = 500$

$$t = \begin{bmatrix} 0700 \\ 0800 \\ 0900 \end{bmatrix} \qquad K(t,t) = \{k(t_i, t_j)\}_{i,j} = \begin{bmatrix} 49.0 & 48.0 & 45.2 \\ 48.0 & 49.0 & 48.0 \\ 45.2 & 48.0 & 49.0 \end{bmatrix}$$

Intuitive summary of GP so far

- GP offer distributions over functions (infinite numbers of jointly Gaussian variables)
- For *any* finite subset vector t, we have a normal distribution:

$$f(t) \sim \mathcal{N}(0, K(t,t))$$

- where covariance matrix K is calculated by plugging t into kernel $k(\cdot,\cdot)$.
- New notation: $f \sim \mathcal{GP}\left(m(\cdot), k(\cdot,\cdot)\right)$ or $f \sim \mathcal{GP}(m,k)$.

Important Gaussian properties (for today's purposes):

- additivity (forming a joint)
- conditioning (inference)
- expectations (posterior and predictive moments)
- marginalization (marginal likelihood/model selection)
- ...

Additivity (joint)

- prior (or latent) $f \sim \mathcal{N}(m_f, K_{ff})$
- additive iid noise $n \sim \mathcal{N}(0, \sigma_n^2 I)$
- let $y = f + n$, then:

$$p(y,f) = p(y|f)p(f) = \mathcal{N}\left(\begin{bmatrix} f \\ y \end{bmatrix}; \begin{bmatrix} m_f \\ m_y \end{bmatrix}, \begin{bmatrix} K_{ff} & K_{fy} \\ K_{fy}^T & K_{yy} \end{bmatrix}\right)$$

- where (in this case):

$$K_{fy} = E[(f - m_f)(y - m_y)^T] = K_{ff} \qquad\qquad K_{yy} = K_{ff} + \sigma_n^2 I$$

- latent f and noisy observation y are jointly Gaussian

Where did the GP go?

- prior (or latent) $f \sim \mathcal{N}(m_f, K_{ff})$
- additive iid noise $n \sim \mathcal{N}(0, \sigma_n^2 I)$
- let $y = f + n$, then:

$$p(y, f) = p(y|f)p(f) = \mathcal{N}\left(\begin{bmatrix} f \\ y \end{bmatrix}; \begin{bmatrix} m_f \\ m_y \end{bmatrix}, \begin{bmatrix} K_{ff} & K_{fy} \\ K_{fy}^T & K_{yy} \end{bmatrix}\right)$$

- If f and y are indexed by some input points t:

$$m_f = \begin{bmatrix} m_f(t_1) \\ \vdots \\ m_f(t_n) \end{bmatrix} \qquad K_{ff} = \{k(t_i, t_j)\}_{i,j=1\ldots n} \qquad \ldots$$

Where did the GP go?

- prior (or latent) $f \sim \mathcal{GP}(m_f, k_{ff})$
- additive iid noise $n \sim \mathcal{GP}(0, \sigma_n^2 \delta)$
- let $y = f + n$, then:

$$p(y(t), f(t)) = p(y|f)p(f) = \mathcal{N}\left(\begin{bmatrix} f \\ y \end{bmatrix}; \begin{bmatrix} m_f \\ m_y \end{bmatrix}, \begin{bmatrix} K_{ff} & K_{fy} \\ K_{fy}^T & K_{yy} \end{bmatrix}\right)$$

- If f and y are indexed by some input points t:

$$m_f = \begin{bmatrix} m_f(t_1) \\ \vdots \\ m_f(t_n) \end{bmatrix} \qquad K_{ff} = \{k(t_i, t_j)\}_{i,j=1\ldots n} \qquad \ldots$$

Conditioning (inference)

- If f and y are jointly Gaussian:

$$\begin{bmatrix} f \\ y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m_f \\ m_y \end{bmatrix}, \begin{bmatrix} K_{ff} & K_{fy} \\ K_{fy}^T & K_{yy} \end{bmatrix}\right)$$

- Then:

$$f|y \sim \mathcal{N}\left(K_{fy}K_{yy}^{-1}(y - m_y) + m_f \quad , \quad K_{ff} - K_{fy}K_{yy}^{-1}K_{fy}^T\right)$$

- inference of latent given data is simple linear algebra.

$$p(f|y) = \frac{p(y|f)p(f)}{p(y)}$$

Expectation (posterior and predictive moments)

- Conditioning on data gave us:

$$f|y \sim \mathcal{N}\left(K_{fy}K_{yy}^{-1}(y - m_y) + m_f \quad , \quad K_{ff} - K_{fy}K_{yy}^{-1}K_{fy}^T\right)$$

- then $E[f|y] = K_{fy}K_{yy}^{-1}(y - m_y) + m_f$ (MAP, posterior mean, ...)
- Predict data observations y^*:

$$\begin{bmatrix} y \\ y^* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m_y \\ m_{y^*} \end{bmatrix}, \begin{bmatrix} K_{yy} & K_{y^*y} \\ K_{y^*y}^T & K_{y^*y^*} \end{bmatrix}\right)$$

- no different:

$$y^*|y \sim \mathcal{N}\left(K_{y^*y}K_{yy}^{-1}(y - m_y) + m_{y^*} \quad , \quad K_{y^*y^*} - K_{y^*y}K_{yy}^{-1}K_{y^*y}^T\right)$$

Marginalization (likelihood and model selection)

- Again, if:

$$\begin{bmatrix} f \\ y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m_f \\ m_y \end{bmatrix}, \begin{bmatrix} K_{ff} & K_{fy} \\ K_{fy}^T & K_{yy} \end{bmatrix}\right)$$

- we can marginalize out the latent:

$$p(y) = \int p(y|f)p(f)df \qquad \leftrightarrow \qquad y \sim \mathcal{N}(m_y, K_{yy})$$

- marginal likelihood of the data (or $\log(p(y))$ data log-likelihood)
- In GP context, actually $p(y|\theta) = p(y|\sigma_f, \sigma_n, \ell)$. This can be the basis of model selection.

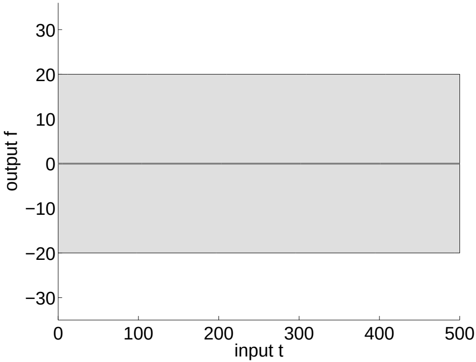
Complaint

- I'm bored. All we are doing is messing around with Gaussians.
- Correct! (sorry)
- This is the whole point.
- We can do some remarkable things...

Outline

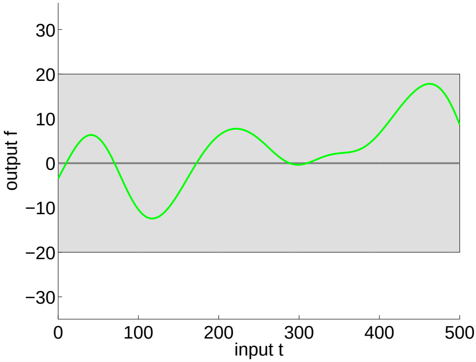
Our example model

- $f \sim \mathcal{GP}(0, k_{ff})$, where $k_{ff}(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$
- $y|f \sim \mathcal{GP}(f, k_{nn})$, where $k_{nn}(t_i, t_j) = \sigma_n^2 \delta(t_i - t_j)$
- $y \sim \mathcal{GP}(0, k_{yy})$, where $k_{yy}(t_i, t_j) = k_{ff}(t_i, t_j) + k_{nn}(t_i, t_j)$
- We choose $\sigma_f = 10$, $\ell = 50$, $\sigma_n = 1$
- The prior on f:



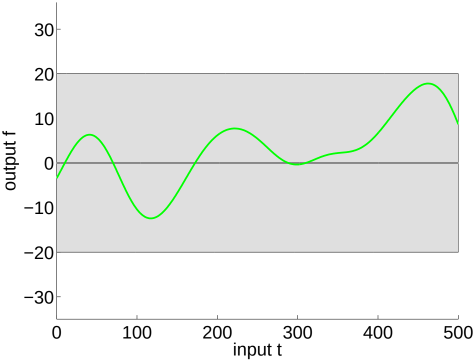
Our example model

- $f \sim \mathcal{GP}(0, k_{ff})$, where $k_{ff}(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$
- $y|f \sim \mathcal{GP}(f, k_{nn})$, where $k_{nn}(t_i, t_j) = \sigma_n^2 \delta(t_i - t_j)$
- $y \sim \mathcal{GP}(0, k_{yy})$, where $k_{yy}(t_i, t_j) = k_{ff}(t_i, t_j) + k_{nn}(t_i, t_j)$
- We choose $\sigma_f = 10$, $\ell = 50$, $\sigma_n = 1$
- A draw from f:



Drawing from the prior

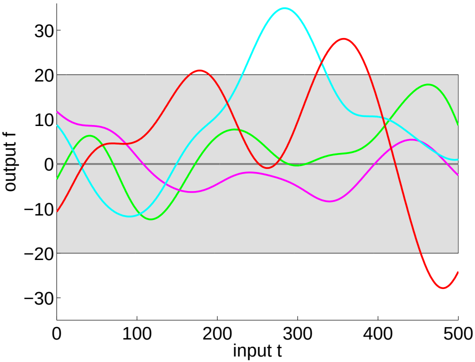
- These steps should be clear:



- Take n (many, but finite!) points $t_i \in [0, 500]$
- Evaluate $K_{ff} = \{k_{ff}(t_i, t_j)\}$
- Draw from $f \sim \mathcal{N}(0, K_{ff})$
- ($\texttt{f} = \texttt{chol}(\texttt{K})' * \texttt{randn}(\texttt{n}, \texttt{1})$)

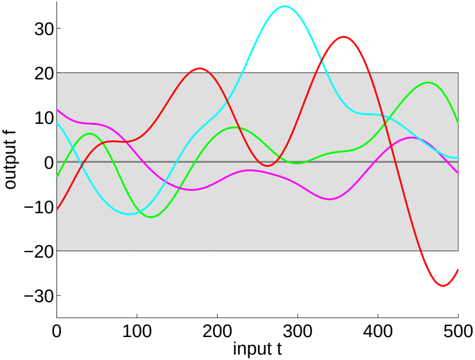
Draw a few more

- four draws from f:



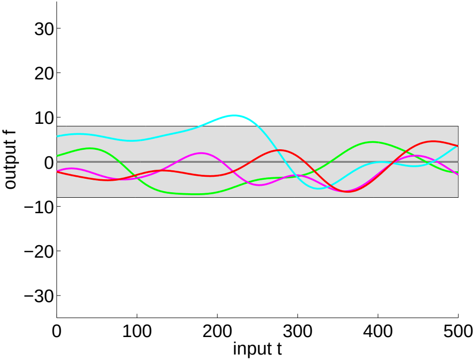
Impact of hyperparameters

- $\sigma_f = 10$, $\ell = 50$



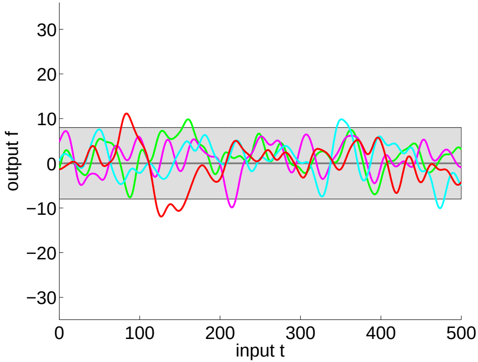
Impact of hyperparameters

- $\sigma_f = 4$, $\ell = 50$



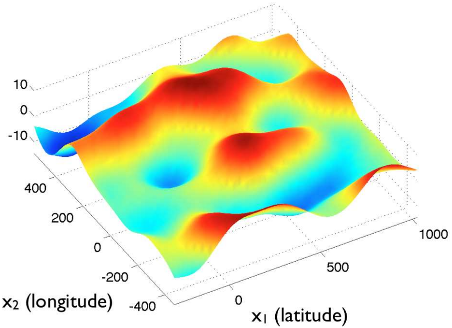
Impact of hyperparameters

- $\sigma_f = 4$, $\ell = 10$



Multidimensional input

- just make each input $x \in \mathbb{R}^D$ (here $D = 2$, *e.g.* lat and long)
- $f \sim \mathcal{GP}(0, k_{ff})$, where $k_{ff}(x^{(i)}, x^{(j)}) = \sigma_f^2 \exp\left\{-\sum_d \frac{1}{2\ell_d^2}(x_d^{(i)} - x_d^{(j)})^2\right\}$



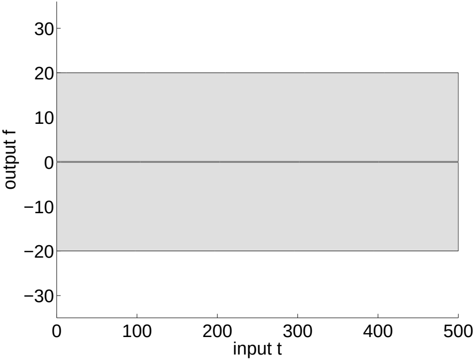
Observations

Same model; we will now gather data y_i.

- $f \sim \mathcal{GP}(0, k_{ff})$, where $k_{ff}(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$
- $y|f \sim \mathcal{GP}(f, k_{nn})$, where $k_{nn}(t_i, t_j) = \sigma_n^2 \delta(t_i - t_j)$
- $y \sim \mathcal{GP}(0, k_{yy})$, where $k_{yy}(t_i, t_j) = k_{ff}(t_i, t_j) + k_{nn}(t_i, t_j)$
- We choose $\sigma_f = 10$, $\ell = 50$, $\sigma_n = 1$

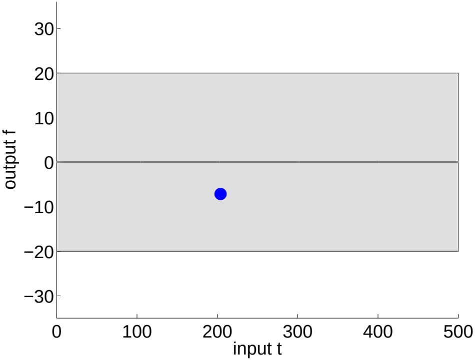
Observations

- the GP prior $p(f)$



Observations

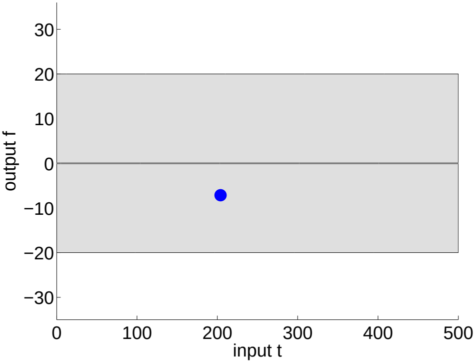
- Observe a single point at $t = 204$:
  $y(204) \sim \mathcal{N}(0, k_{yy}(204, 204)) = \mathcal{N}(0, \sigma_f^2 + \sigma_n^2)$



Observations

- Use conditioning to update the posterior:

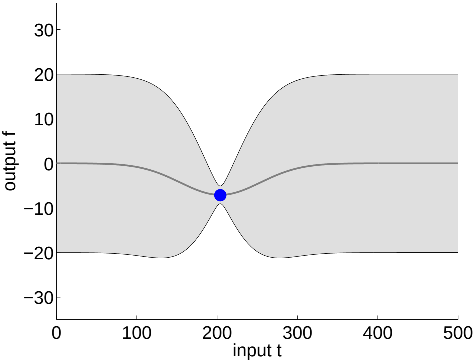
$$f|y(204) \sim \mathcal{N}\left(K_{fy}K_{yy}^{-1}(y(204)-m_y)\;\;,\;\;K_{ff}-K_{fy}K_{yy}^{-1}K_{fy}^{T}\right)$$



Observations

- Use conditioning to update the posterior:

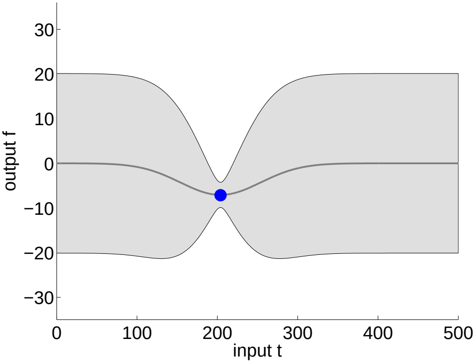
$$f|y(204) \sim \mathcal{N}\left(K_{fy}K_{yy}^{-1}(y(204) - m_y)\ \ ,\ \ K_{ff} - K_{fy}K_{yy}^{-1}K_{fy}^{T}\right)$$



Observations

- ... and the predictive distribution:

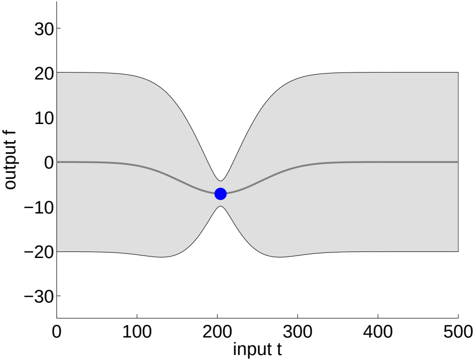
$$y^*|y(204) \sim \mathcal{N}\left(K_{y^*y}K_{yy}^{-1}(y(204) - m_y) \;\;,\;\; K_{y^*y^*} - K_{y^*y}K_{yy}^{-1}K_{y^*y}^{T}\right)$$



Observations

- More observations (data vector y):

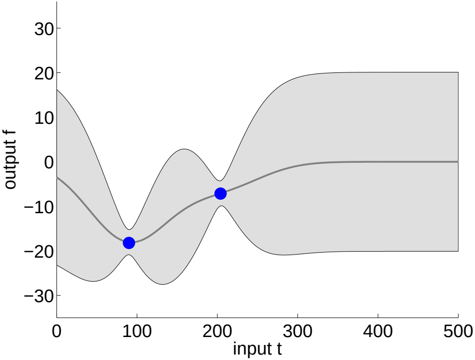
$$y^*|y(\begin{bmatrix}204\\90\end{bmatrix}) \sim \mathcal{N}\left(K_{y^*y}K_{yy}^{-1}\left(y(\begin{bmatrix}204\\90\end{bmatrix}) - m_y\right), K_{y^*y^*} - K_{y^*y}K_{yy}^{-1}K_{y^*y}^T\right)$$



Observations

- More observations (data vector y):

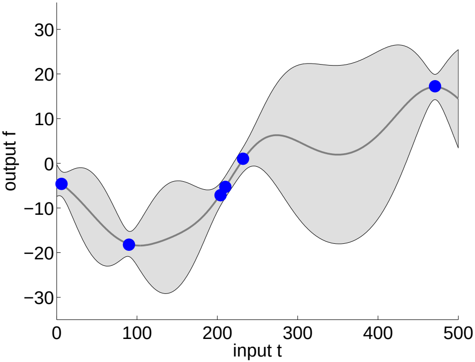
$$y^*|y(\begin{bmatrix}204\\90\end{bmatrix}) \sim \mathcal{N}\left(K_{y^*y}K_{yy}^{-1}\left(y(\begin{bmatrix}204\\90\end{bmatrix}) - m_y\right), K_{y^*y^*} - K_{y^*y}K_{yy}^{-1}K_{y^*y}^T\right)$$



Observations

- More observations (data vector y):

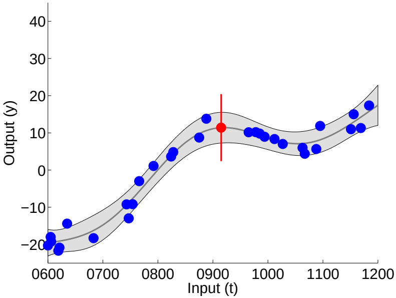
$$y^*|y \sim \mathcal{N}\left(K_{y^*y}K_{yy}^{-1}(y - m_y)\ \ ,\ \ K_{y^*y^*} - K_{y^*y}K_{yy}^{-1}K_{y^*y}^T\right)$$



Observations

- More observations (data vector y):

$$y^*|y \sim \mathcal{N}\left(K_{y^*y}K_{yy}^{-1}(y - m_y)\ \ ,\ \ K_{y^*y^*} - K_{y^*y}K_{yy}^{-1}K_{y^*y}^T\right)$$



Nonparametric Regression

- GP let the data speak for itself... but all the data must speak.

$$y^*|y \sim \mathcal{N}\left(K_{y^*y}K_{yy}^{-1}(y - m_y) \;\;,\;\; K_{y^*y^*} - K_{y^*y}K_{yy}^{-1}K_{y^*y}^T\right)$$

- “nonparametric models have an infinite number of parameters”

Nonparametric Regression

- GP let the data speak for itself... but all the data must speak.

$$y^*|y \sim \mathcal{N}\left(K_{y^*y}K_{yy}^{-1}(y - m_y)\ \ ,\ \ K_{y^*y^*} - K_{y^*y}K_{yy}^{-1}K_{y^*y}^T\right)$$

- ~~“nonparametric models have an infinite number of parameters”~~

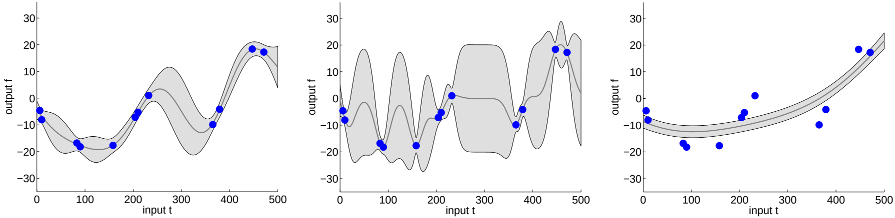
- “nonparametric models have a finite but unbounded number of parameters that grows with data”

Almost through the basics...

Model Selection / Hyperparameter Learning

- $f \sim \mathcal{GP}(0, k_{ff})$, where $k_{ff}(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$

$\ell = 50$: just right $\quad$ $\ell = 15$: overfitting $\quad$ $\ell = 250$: underfitting



Model Selection (1): Marginal Likelihood

- $p(y) = \mathcal{N}(0, K_{yy}) \quad \rightarrow \quad p(y|\sigma_f, \sigma_n, \ell) = \mathcal{N}\left(0, K_{yy}(\sigma_f, \sigma_n, \ell)\right)$
- not obvious why this should not over (or under) fit, but it’s in the math...

$$\log\left(p(y|\sigma_f, \sigma_n, \ell)\right) = -\frac{1}{2}y^T K_{yy}^{-1} y - \frac{1}{2}\log|K_{yy}| - \frac{n}{2}\log(2\pi)$$

- “Occam’s Razor” via regularization/probabilistic model
- (how do the parameters trade off against each other here?)

Model Selection (2): Cross Validation

- Can also consider predictive distribution for some held out data:

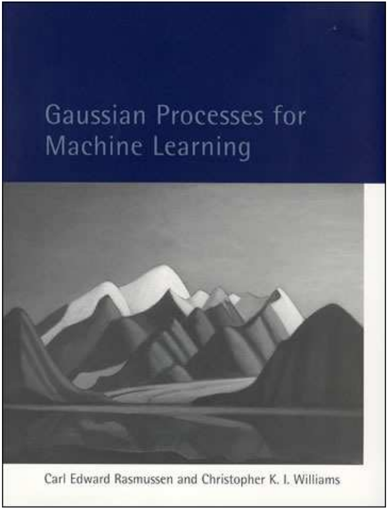
$$PL(\sigma_f, \sigma_n, \ell) = \log\left(p(y_{\text{test}} | y_{\text{train}}, \sigma_f, \sigma_n, \ell)\right)$$

- Again a Gaussian.
- Again can take derivatives and tune model hyperparameters.

Outline

More details

- GP basics: Appdx A, Ch 1
- Regression: Ch 2
- Kernels: Ch 4
- Model Selection: Ch 5



What's next?

- Revisit the model and see what can be hacked:

$$f \sim \mathcal{GP}(0, k_{ff}), \quad \text{where} \;\; k_{ff}(t_i, t_j) = \sigma_f^2 \exp\left\{ -\frac{1}{2\ell^2}(t_i - t_j)^2 \right\}$$

$$y_i | f_i \sim \mathcal{N}(f_i, \sigma_n^2 I)$$

-
-
-
-

What's next?

- Revisit the model and see what can be hacked:

$$f \sim \mathcal{GP}(0, k_{ff}), \quad \text{where} \;\; k_{ff}(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$$

$$y_i | f_i \sim \mathcal{N}(f_i, \sigma_n^2 I)$$

- Option 1: hyperparameters $\rightarrow$ model selection.
-
-
-

What's next?

- Revisit the model and see what can be hacked:

$$f \sim \mathcal{GP}(0, k_{ff}), \quad \text{where} \;\; k_{ff}(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$$

$$y_i | f_i \sim \mathcal{N}(f_i, \sigma_n^2 I)$$

- Option 1: hyperparameters $\rightarrow$ model selection.
- Option 2: functional form of k_{ff} $\rightarrow$ kernel choices.
-
-

What's next?

- Revisit the model and see what can be hacked:

$$f \sim \mathcal{GP}(0, k_{ff}), \quad \text{where } k_{ff}(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$$

$$y_i | f_i \sim \mathcal{N}(f_i, \sigma_n^2 I)$$

- Option 1: hyperparameters $\rightarrow$ model selection.
- Option 2: functional form of k_{ff} $\rightarrow$ kernel choices.
- Option 3: computation
-

What's next?

- Revisit the model and see what can be hacked:

$$f \sim \mathcal{GP}(0, k_{ff}), \quad \text{where} \;\; k_{ff}(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$$

$$y_i | f_i \sim \mathcal{N}(f_i, \sigma_n^2 I)$$

- Option 1: hyperparameters $\rightarrow$ model selection.
- Option 2: functional form of k_{ff} $\rightarrow$ kernel choices.
- Option 3: computation
- Option 4: the data distribution $\rightarrow$ likelihood choices.

Outline

What's next?

- Revisit the model and see what can be hacked:

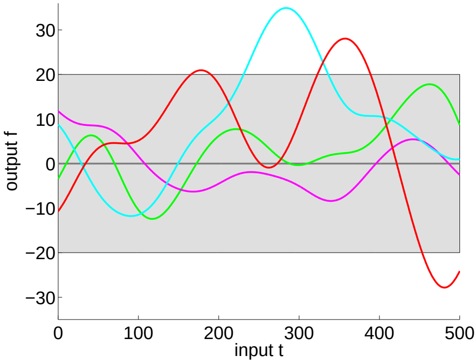
$$f \sim \mathcal{GP}(0, k_{ff}), \quad \text{where } k_{ff}(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$$

$$y_i | f_i \sim \mathcal{N}(f_i, \sigma_n^2 I)$$

- Option 1: hyperparameters $\rightarrow$ model selection.
- Option 2: functional form of k_{ff} $\rightarrow$ kernel choices.
- Option 3: computation
- Option 4: the data distribution $\rightarrow$ likelihood choices.

What the kernel is doing (SE)

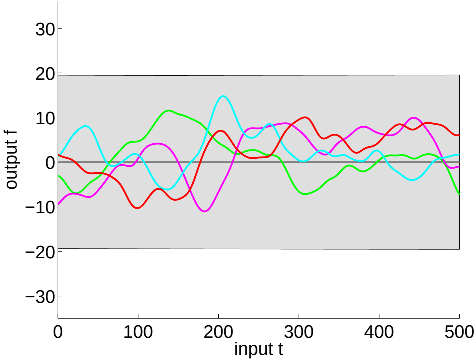
$$k(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$$



Rational Quadratic

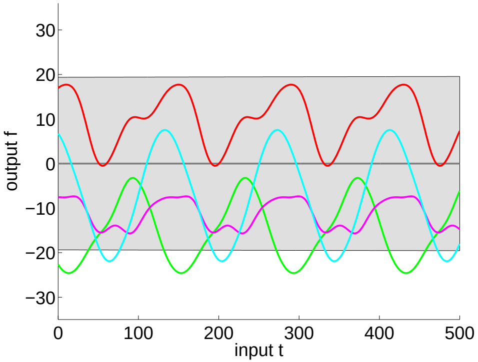
$$k(t_i, t_j) = \sigma_f^2 \left(1 + \frac{1}{2\alpha\ell^2}(t_i - t_j)^2\right)^{-\alpha}$$

$$\propto \quad \sigma_f^2 \int z^{\alpha-1} \exp\left(-\frac{\alpha z}{\beta}\right) \exp\left(-\frac{z(t_i - t_j)^2}{2}\right) dz$$



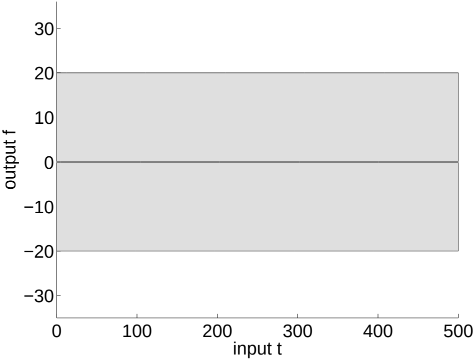
Periodic

$$k(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{2}{\ell^2}\sin^2\left(\frac{\pi}{p}|t_i - t_j|\right)\right\}$$



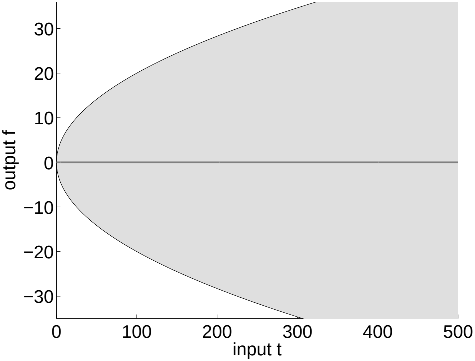
From Stationary to Nonstationary Kernels

- $k(t_i, t_j) = k(t_i - t_j) = k(\tau)$
- $k(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$



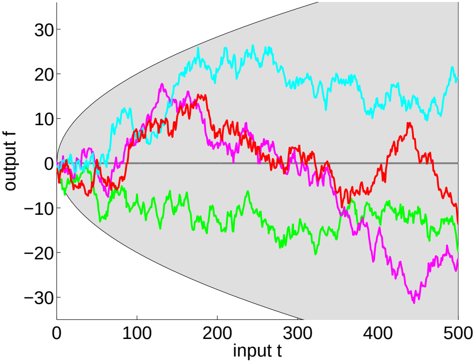
Wiener Process

- $k(t_i, t_j) = \min(t_i, t_j)$
- Still a GP



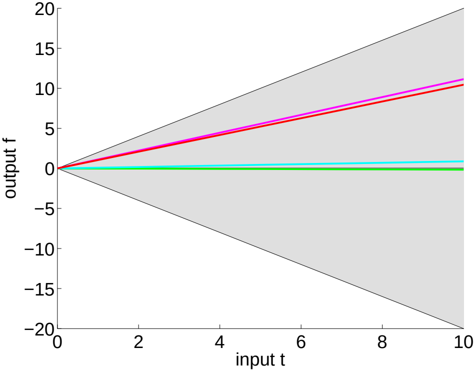
Wiener Process

- $k(t_i, t_j) = \min(t_i, t_j)$
- Draws from a nonstationary GP



Linear Regression...

- $f(t) = wt$ with $w \sim \mathcal{N}(0, 1)$
- $k(t_i, t_j) = E[f(t_i)f(t_j)] = t_i t_j$



Build your own kernel (1): Operations

- Linear: $k(t_i, t_j) = \alpha k_1(t_i, t_j) + \beta k_2(t_i, t_j)$ (for $\alpha, \beta \geq 0$)

  or $k\left(x^{(i)}, x^{(j)}\right) = k_a\left(x_1^{(i)}, x_1^{(j)}\right) + k_b\left(x_2^{(i)}, x_2^{(j)}\right)$

- Products: $k(t_i, t_j) = k_1(t_i, t_j)k_2(t_i, t_j)$

- Integration: $z(t) = \int g(u, t)f(u)du \quad \leftrightarrow$

  $k_z(t_i, t_j) = \int\int g(u, t_1)k_f(t_i, t_j)g(v, t_j)dudv$

- Differentiation: $z(t) = \frac{\partial}{\partial t}f(t) \quad \leftrightarrow \quad k_z(t_i, t_j) = \frac{\partial^2}{\partial t_i \partial t_j}k_f(t_i, t_j)$

- Warping: $z(t) = f\left(h(t)\right) \quad \leftrightarrow \quad k_z(t_i, t_j) = k_f\left(h(t_i), h(t_j)\right)$

Preserves joint Gaussianity (mostly)!

- Linear: $k(t_i, t_j) = \alpha k_1(t_i, t_j) + \beta k_2(t_i, t_j)$

  or $k\left(x^{(i)}, x^{(j)}\right) = k_a\left(x_1^{(i)}, x_1^{(j)}\right) + k_b\left(x_2^{(i)}, x_2^{(j)}\right)$
- Products: $k(t_i, t_j) = k_1(t_i, t_j)k_2(t_i, t_j)$
- Integration: $z(t) = \int g(u, t)f(u)du \quad \leftrightarrow$

  $k_z(t_i, t_j) = \int \int g(u, t_1)k_f(t_i, t_j)g(v, t_j)dudv$
- Differentiation: $z(t) = \frac{\partial}{\partial t}f(t) \quad \leftrightarrow \quad k_z(t_i, t_j) = \frac{\partial^2}{\partial t_i \partial t_j}k_f(t_i, t_j)$
- Warping: $z(t) = f\left(h(t)\right) \quad \leftrightarrow \quad k_z(t_i, t_j) = k_f\left(h(t_i), h(t_j)\right)$

Build your own kernel (2): frequency domain

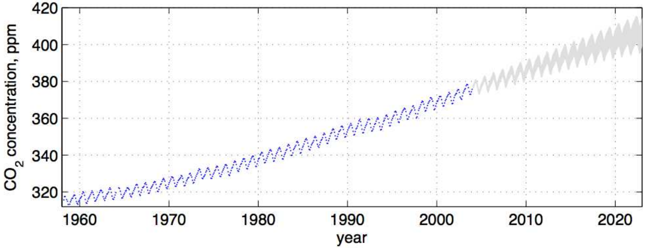
- a *stationary* kernel $k(t_i, t_j) = k(t_i - t_j) = k(\tau)$ is positive semidefinite (satisfies Mercer) iff:

$$S(\omega) = \mathcal{F}\{k\}(\omega) \geq 0 \ \ \forall \ \ \omega$$

- Power spectral density (Wiener-Khinchin, ...)
- Note $k(0) = \int S(\omega) d\omega$

Kernel Summary

- GP gives a distribution over functions...
- the kernel determines the type of functions.
- can/should be tailored to application
- toward a GP toolbox



Outline

What's next?

- Revisit the model and see what can be hacked:

$$f \sim \mathcal{GP}(0, k_{ff}), \quad \text{where} \quad k_{ff}(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$$

$$y_i | f_i \sim \mathcal{N}(f_i, \sigma_n^2 I)$$

- Option 1: hyperparameters $\rightarrow$ model selection.
- Option 2: functional form of k_{ff} $\rightarrow$ kernel choices.
- Option 3: computation
- Option 4: the data distribution $\rightarrow$ likelihood choices.

Sounds great, but...

Nonparametric flexibility

- ... but we have to compute on all the data:

$$f|y \sim \mathcal{N}\left(K_{fy}K_{yy}^{-1}(y-m_y)+m_f \quad , \quad K_{ff}-K_{fy}K_{yy}^{-1}K_{fy}^T\right)$$

$$\log\left(p(y|\sigma_f,\sigma_n,\ell)\right) = -\frac{1}{2}y^T K_{yy}^{-1} y - \frac{1}{2}\log|K_{yy}| - \frac{n}{2}\log(2\pi)$$

- What does this cost?
- $\mathcal{O}(n^3)$ in runtime, $\mathcal{O}(n^2)$ in memory
- When can I simplify?
- special structure methods (kernels, input points, etc.)
- sparsification methods (pseudo points, etc.)

What's next?

- Revisit the model and see what can be hacked:

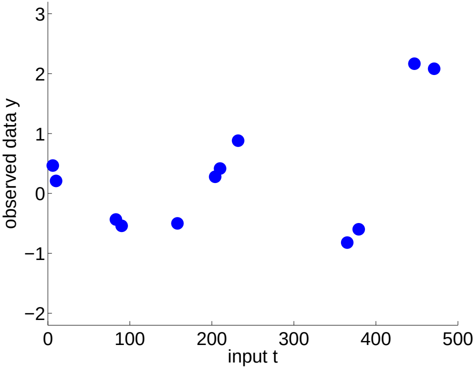
$$f \sim \mathcal{GP}(0, k_{ff}), \quad \text{where } \; k_{ff}(t_i, t_j) = \sigma_f^2 \exp\left\{-\frac{1}{2\ell^2}(t_i - t_j)^2\right\}$$

$$y_i | f_i \sim \mathcal{N}(f_i, \sigma_n^2 I)$$

- Option 1: hyperparameters $\rightarrow$ model selection.
- Option 2: functional form of k_{ff} $\rightarrow$ kernel choices.
- Option 3: computation
- Option 4: the data distribution $\rightarrow$ likelihood choices.

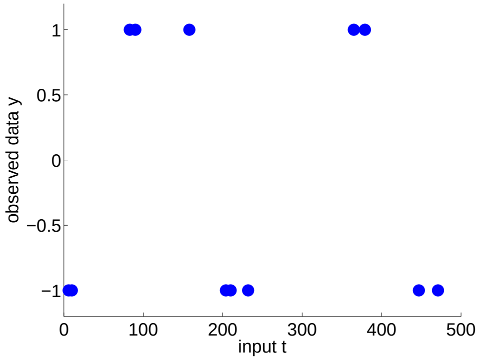
Data up to now

- continuous regression made sense
- data likelihood model: $y_i|f_i \sim \mathcal{N}(f_i, \sigma_n^2 I)$



Binary label data

- Classification (not regression) setting
- $y_i|f_i \sim \mathcal{N}(f_i, \sigma_n^2 I)$ is inappropriate

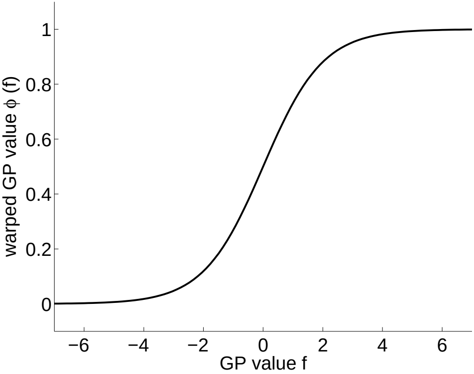


GP Classification

- Probit or Logistic "regression" model on $y_i \in \{-1, +1\}$:

$$p(y_i|f_i) = \phi(y_i f_i) = \frac{1}{1 + \exp(-y_i f_i)}$$

- Warps f onto the $[0, 1]$ interval

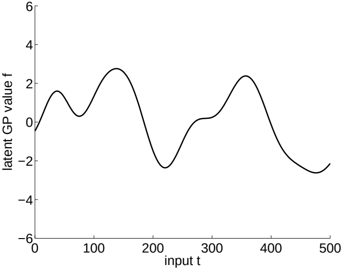


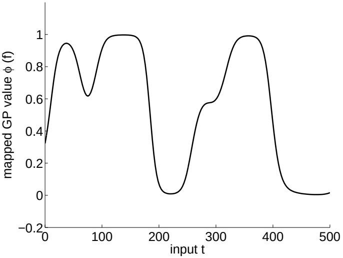
GP Classification

- Probit or Logistic "regression" model on $y_i \in \{-1, +1\}$:

$$p(y_i|f_i) = \phi(y_i f_i) = \frac{1}{1 + \exp(-y_i f_i)}$$

- Warps f onto the $[0, 1]$ interval





What we want to calculate

- predictive distribution:

$$p(y^*|y) = \int p(y^*|f^*)p(f^*|y)df^*$$

- predictive posterior:

$$p(f^*|y) = \int p(f^*|f)p(f|y)df$$

- data posterior:

$$p(f|y) = \frac{\prod_i p(y_i|f_i)p(f)}{p(y)}$$

- *None* of which is tractable to compute

However...

- predictive distribution:

$$p(y^*|y) = \int p(y^*|f^*)q(f^*|y)df^*$$

- predictive posterior:

$$q(f^*|y) = \int p(f^*|f)q(f|y)df$$

- data posterior:

$$q(f|y) \approx p(f|y) = \frac{\prod_i p(y_i|f_i)p(f)}{p(y)}$$

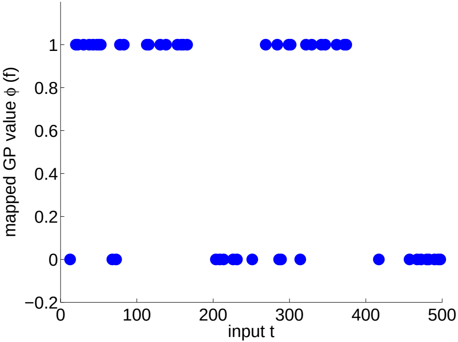
- If q is Gaussian, these are tractable to compute

Approximate Inference

- Methods for producing a Gaussian $q(f|y) \approx p(f|y)$
- Laplace Approximation, Expectation Propagation, Variational Inference
- Technologies within a GP method
- Subject of much research; often work well

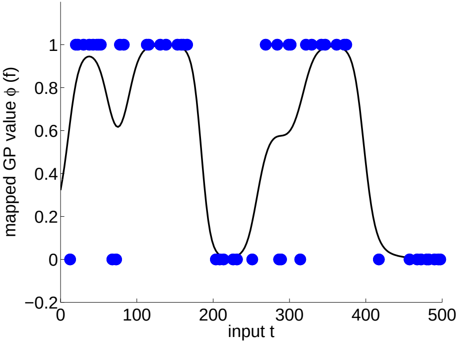
Using Approximate Inference

- Allows "regression" on the $[0, 1]$ interval



Using Approximate Inference

- Allows "regression" on the $[0, 1]$ interval



Outline

Conclusions

- Gaussian Processes can be effective tools for regression and classification
- Quantified uncertainty can be highly valuable
- GP can be extended in interesting ways (linearity helps)
- GP appear as limits or general cases of a number of ML technologies
- GP are not without problems

Some References/Pointers/Credits

- Rasmussen and Williams, *Gaussian Processes for Machine Learning*
- Bishop, *Pattern Recognition and Machine Learning*
- www.gaussianprocess.org (better updated/kept than .com)
- loads of papers at AISTATS/NIPS/ICML/JMLR over the last 12 years.

Outline

What about SVM?

- illustrate flexibility of GP
- draw an interesting connection
- GP joint:

$$-\log p(y, f) = \frac{1}{2} f^T K_{ff}^{-1} f - \sum_i \log(p(y_i|f_i))$$

- SVM loss (for $f_i = f(x_i) = w^T x_i$):

$$\ell(w) = \frac{1}{2} w^T w + C \sum_i (1 - y_i f_i)$$

SVM

- illustrate flexibility of GP
- draw an interesting connection
- GP joint:

$$-\log p(y, f) = \frac{1}{2} f^T K^{-1} f - \sum_i \log(p(y_i|f_i))$$

- SVM loss (for $f_i = f(x_i) = \phi(x_i) = k(\cdot, x_i)$):

$$\ell(\phi) = \frac{1}{2} f^T K^{-1} f + C \sum_i (1 - y_i f_i)$$

- (more reading: Seeger (2002), Relationship between GP, SVM, Splines)

Connections

Already we have seen:

- Wiener processes
- linear regression
- SVM
- what else?

Temporal linear Gaussian models

- Wiener process (Brownian motion, random walk, OU process)
- Linear dynamical system (state space model, Kalman filter/smoother, etc.)

$$f(t) = Af(t-1) + w(t) \qquad\qquad y(t) = f(t) + n(t)$$

- Gauss-Markov processes (ARMA(p,q), etc.)

$$f(t) = \sum_{i=1}^{p} \alpha_i f(t-i) + \sum_{i=1}^{q} \beta_i w(t-i)$$

- Intuition of linearity and Gaussianity $\rightarrow$ GP

Other nonparametric models (or parametric limits)

- Kernel smoothing (Nadaraya-Watson, locally weighted regression):

$$y^* = \sum_{i=1}^{n} \alpha_i y_i \quad \text{where} \quad \alpha_i = k(t_i, t^*)$$

- Compare to:

$$y^*|y \sim \mathcal{N}\left(K_{y^*y}K_{yy}^{-1}(y - m_y) \ , \ K_{y^*y^*} - K_{y^*y}K_{yy}^{-1}K_{y^*y}^{T}\right)$$

- Neural network limit (infinite bases, important to know about, Neal ’96):

$$f(t) = \lim_{m\to\infty} \sum_{i=1}^{m} v_i h(t; u_i)$$