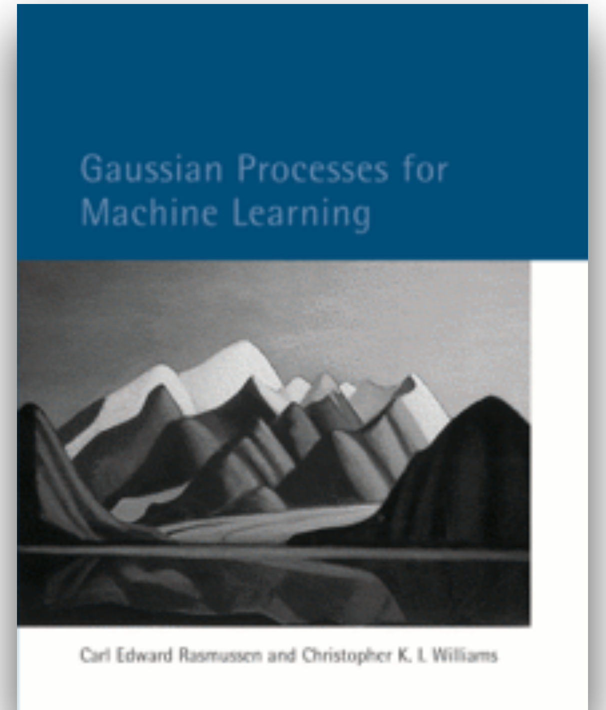
6.867
Gaussian Processes

Fall 2016

Historical Perspective

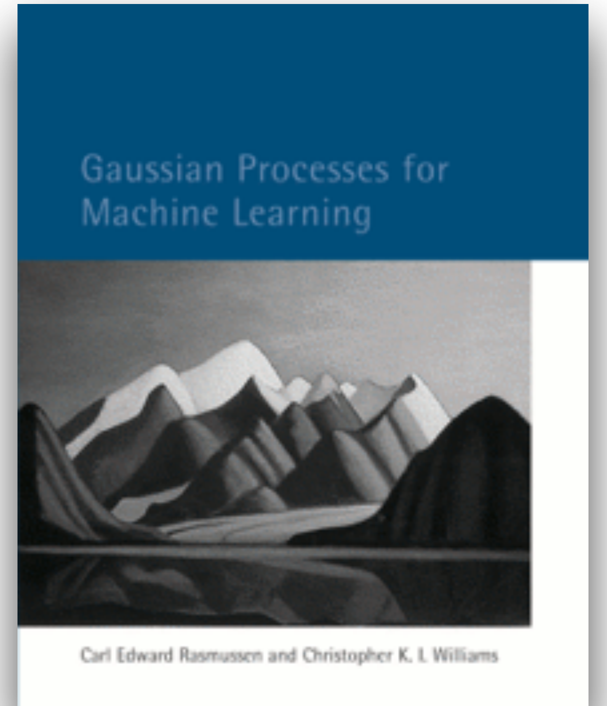


(2006, MIT Press)

- Carl & Chris, 1994, Hinton's lab
- NN maturity, connections to physics, probstat
- Time when kernel methods becoming popular

"Many researchers were realizing that neural networks were not so easy to apply in practice, due to the many decisions which needed to be made: what architecture, what activation functions, what learning rate, etc., and the lack of a principled framework to answer these questions…"

Historical Perspective



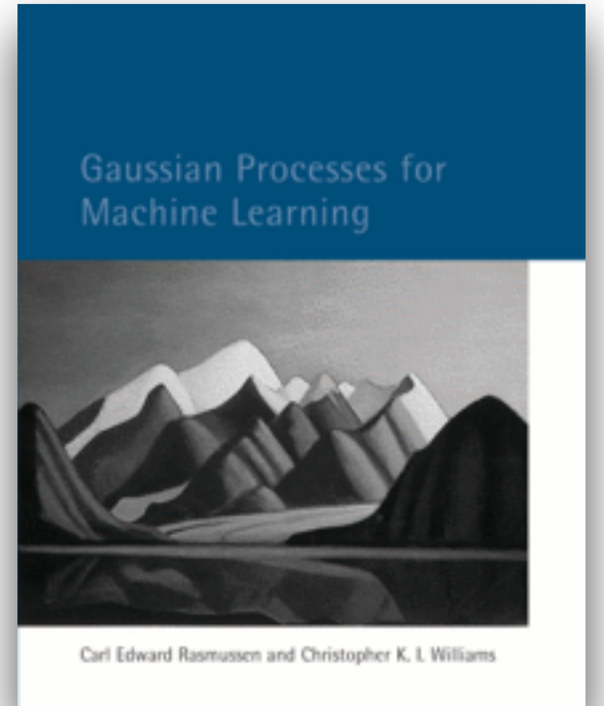
(2006, MIT Press)

- Neal grad student at same lab
- Pursuing probabilistic thinking
- Using Bayesian formalism to avoid "overfitting" when models get large
- Advocated pursuing limits of large models

Neal: Neural networks of infinite size became Gaussian Processes!

(Suggesting possibly simpler inference)

Wider historical perspective

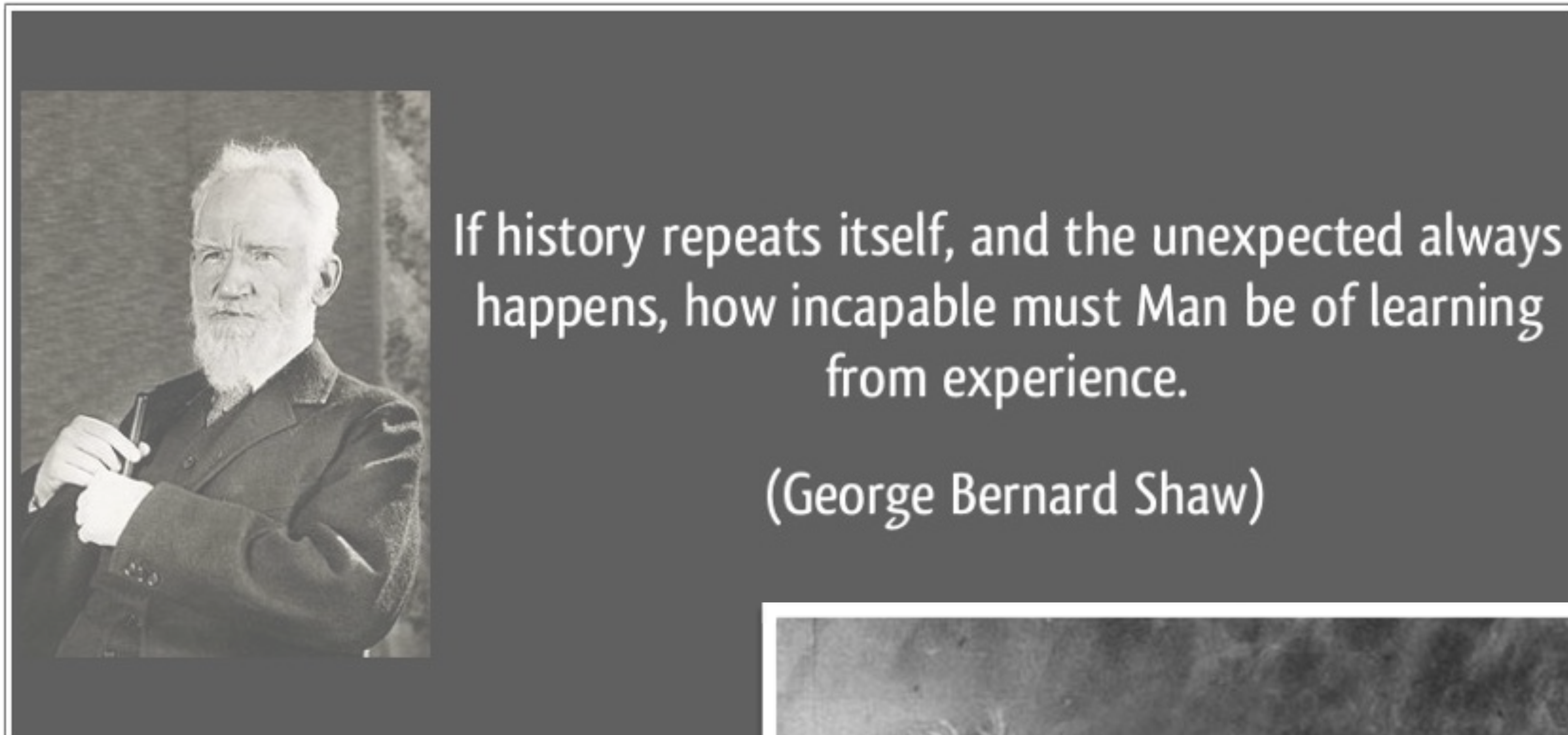


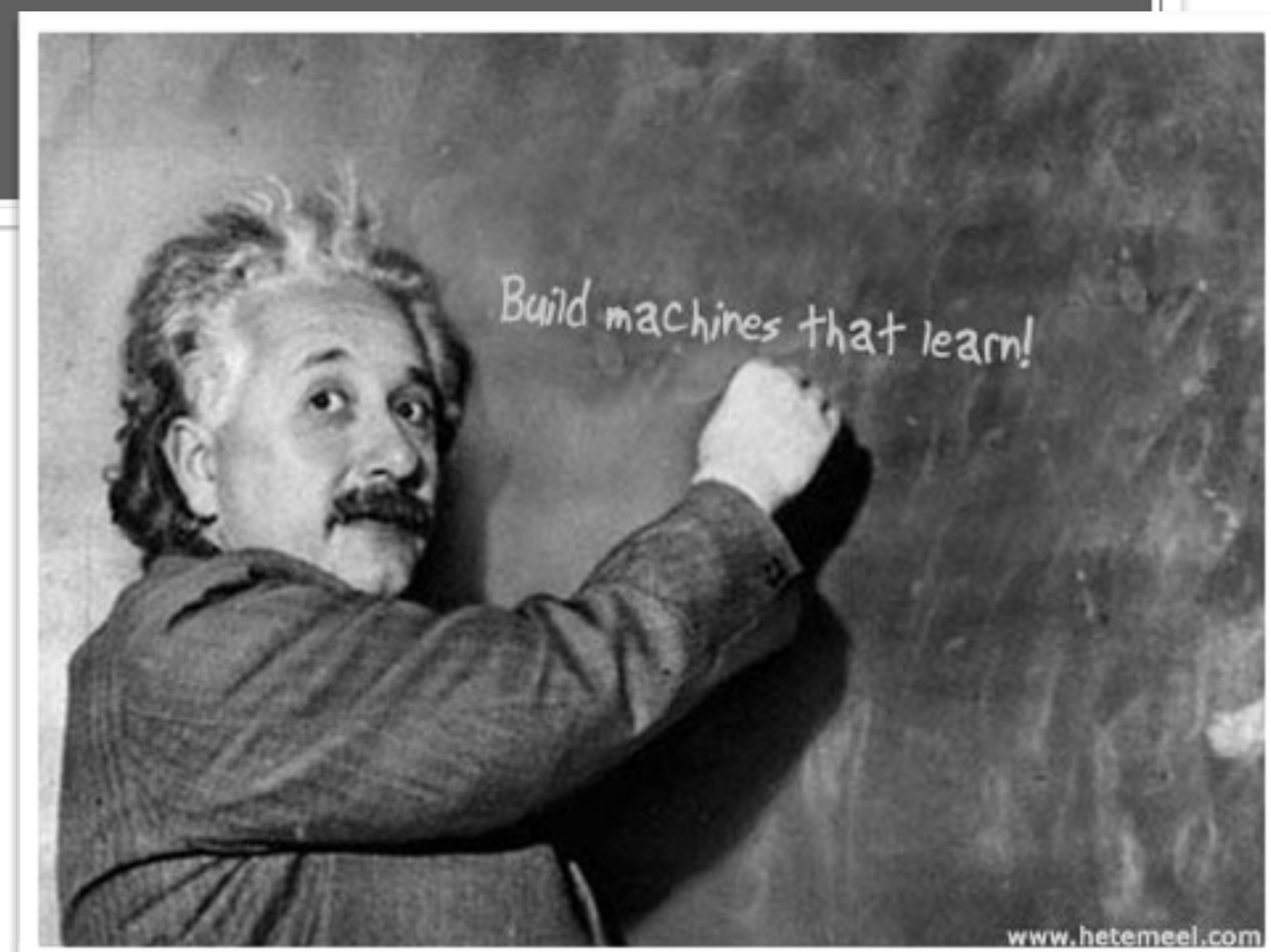
(2006, MIT Press)

- Main reason why NN became popular: *adaptive basis functions*
- Kernels use fixed basis, but allow infinitely many, so fixedness not an issue
- Resulting models much easier to handle than NN (convex), work as well or better

"Thus, one could claim that (as far a machine learning is concerned) the adaptive basis functions were merely a decade-long digression, and we are now back to where we came from. This view is perhaps reasonable if we think of models for solving practical learning problems,…"

though, kernels don't give hidden representations (McKay worried about this in 2003)


If history repeats itself, and the unexpected always happens, how incapable must Man be of learning from experience.
(George Bernard Shaw)

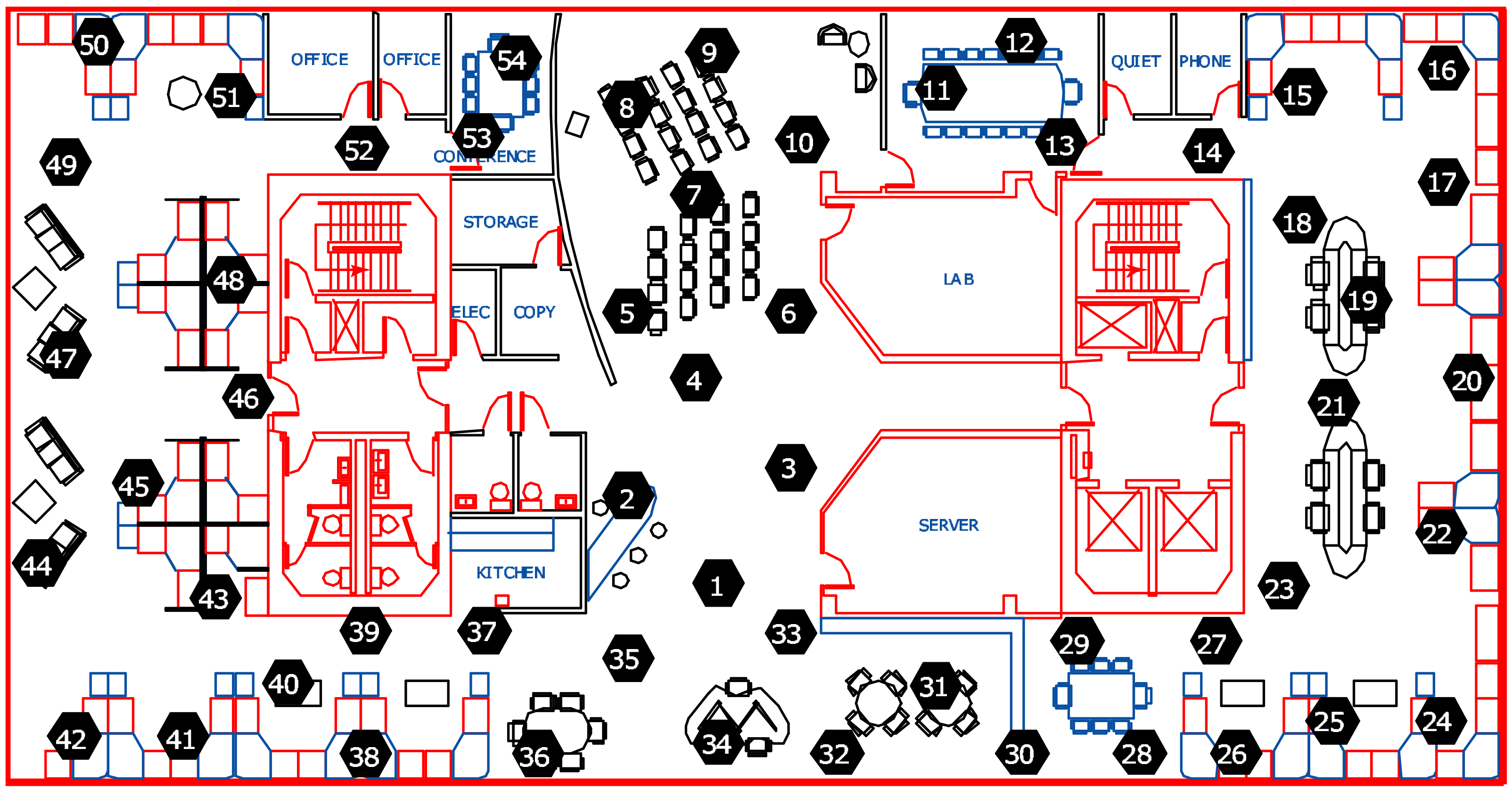

Build machines that learn!
www.hetemeel.com

Aim: Learn a function

Key idea: assume unknown function sampled from a GP

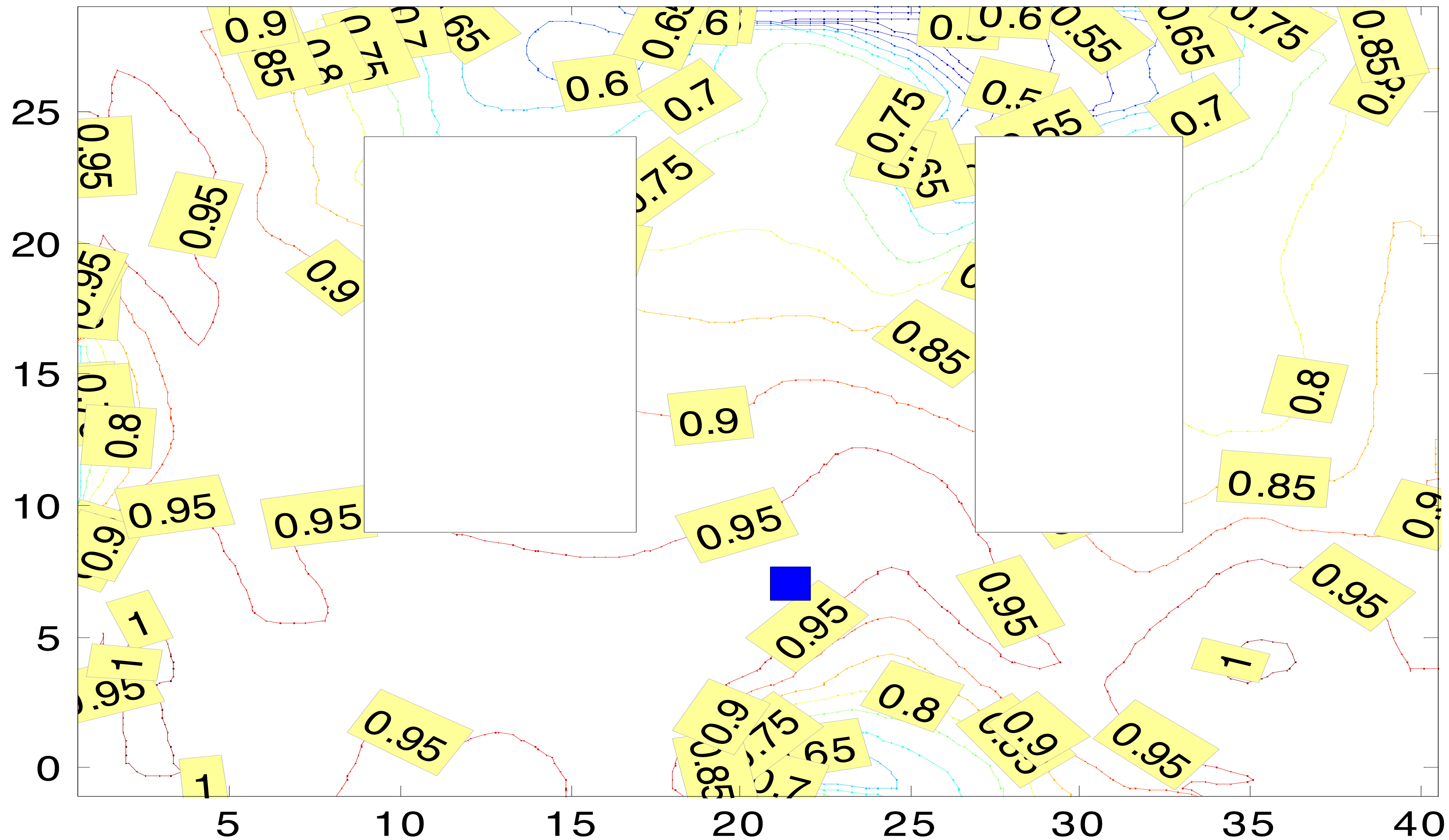
Think of Gaussian Process (GP) as a probability distribution over space of functions!

Modeling temperature distribution



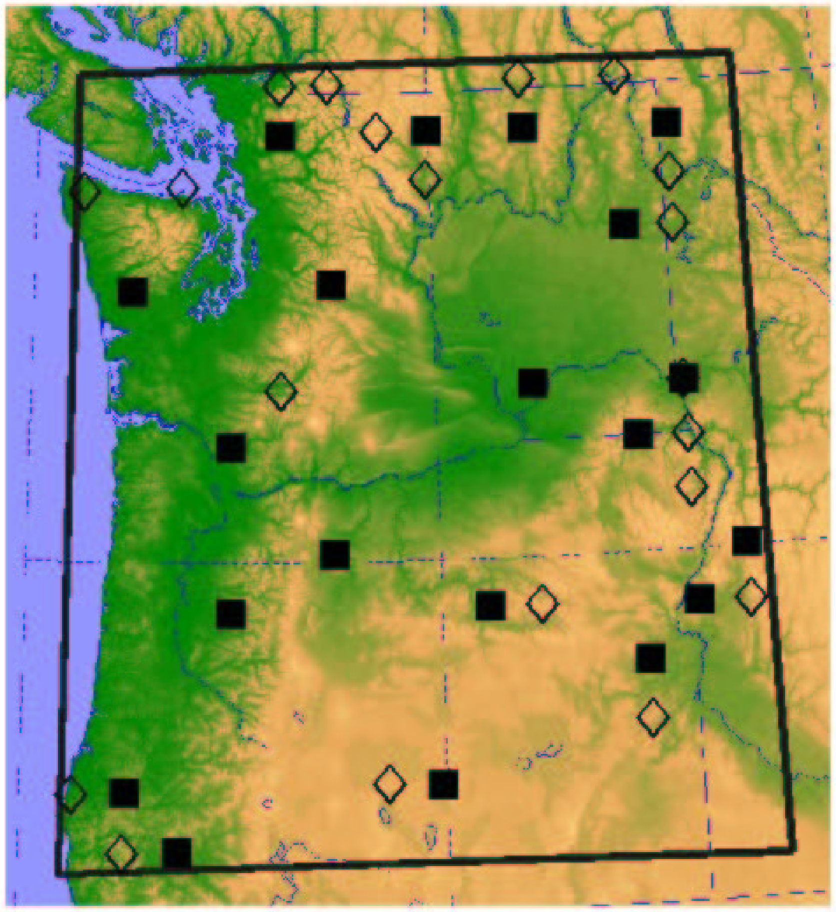
[Krause, Singh, Guestrin, 2008]

Modeling temperature distribution



[Krause, Singh, Guestrin, 2008]

Modeling spatially varying signals



Rain sensors (Pacific NW)

[Krause, Singh, Guestrin, 2008]

Applications of GPs

- Predictive soil modeling
- Geostatistics, Kriging (gold finding!)
- Medical imaging
- Dimensionality reduction, data visualization
- Bayesian optimization
- Automating hyperparameter tuning
- Inverse kinematics (Robotics)
- Probabilistic numerics
- Many more…

Gaussian cheatsheet

- products of Gaussians are Gaussians

$$\mathcal{N}(x; a, A)\mathcal{N}(x; b, B) = \mathcal{N}(x; c, C)\mathcal{N}(a; b, A + B)$$

$$C := (A^{-1} + B^{-1})^{-1} \qquad c := C(A^{-1}a + B^{-1}b)$$

- marginals of Gaussians are Gaussians

$$\int \mathcal{N}\left[\begin{pmatrix} x \\ y \end{pmatrix}; \begin{pmatrix} \mu_x \\ \mu_y \end{pmatrix}, \begin{pmatrix} \Sigma_{xx} & \Sigma_{xy} \\ \Sigma_{yx} & \Sigma_{yy} \end{pmatrix}\right] \mathrm{d}y = \mathcal{N}(x; \mu_x, \Sigma_{xx})$$

- (linear) conditionals of Gaussians are Gaussians

$$p(x \mid y) = \frac{p(x, y)}{p(y)} = \mathcal{N}\left(x; \mu_x + \Sigma_{xy}\Sigma_{yy}^{-1}(y - \mu_y), \Sigma_{xx} - \Sigma_{xy}\Sigma_{yy}^{-1}\Sigma_{yx}\right)$$

- linear projections of Gaussians are Gaussians

$$p(z) = \mathcal{N}(z; \mu, \Sigma) \quad \Rightarrow \quad p(Az) = \mathcal{N}(Az, A\mu, A\Sigma A^{\top})$$

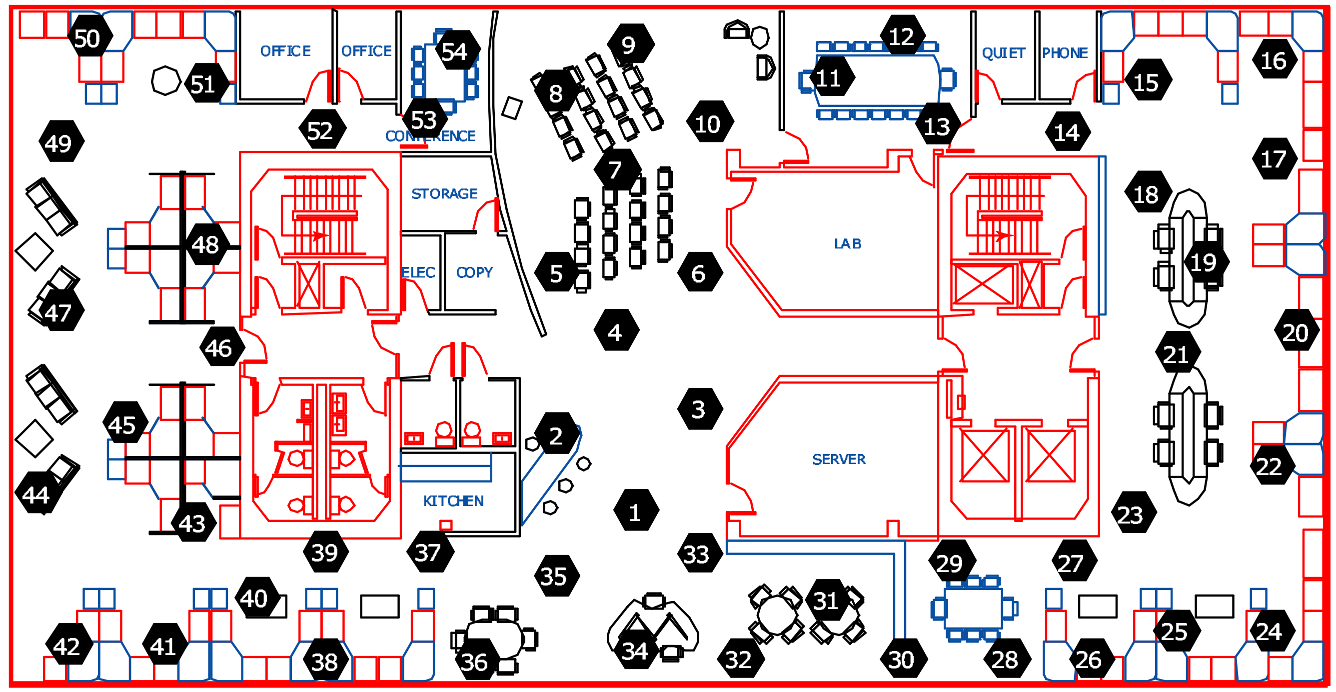
→ Bayesian inference under linear operations

$$p(x) = \mathcal{N}(x; \mu, \Sigma) \qquad p(y \mid x) = \mathcal{N}(y; A^{\top}x + b, \Lambda)$$

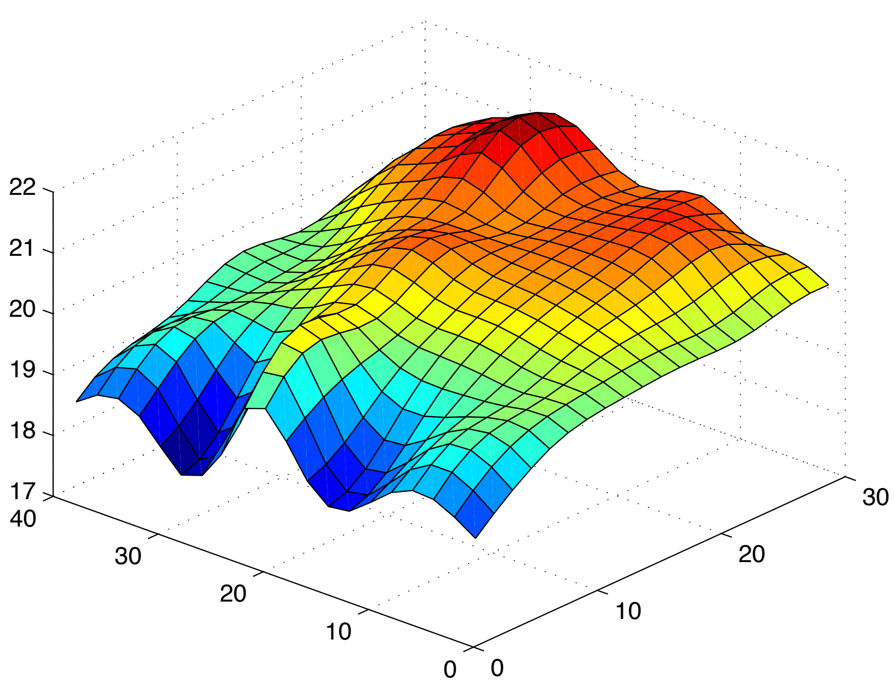
$$p(B^{\top}x + c \mid y) = \mathcal{N}[B^{\top}x + c; B^{\top}\mu + c + B^{\top}\Sigma A(A^{\top}\Sigma A + \Lambda)^{-1}(y - A^{\top}\mu - b),$$
$$B^{\top}\Sigma B - B^{\top}\Sigma A(A^{\top}\Sigma A + \Lambda)^{-1}A^{\top}\Sigma B]$$

Taken from: P. Henning, 2015.

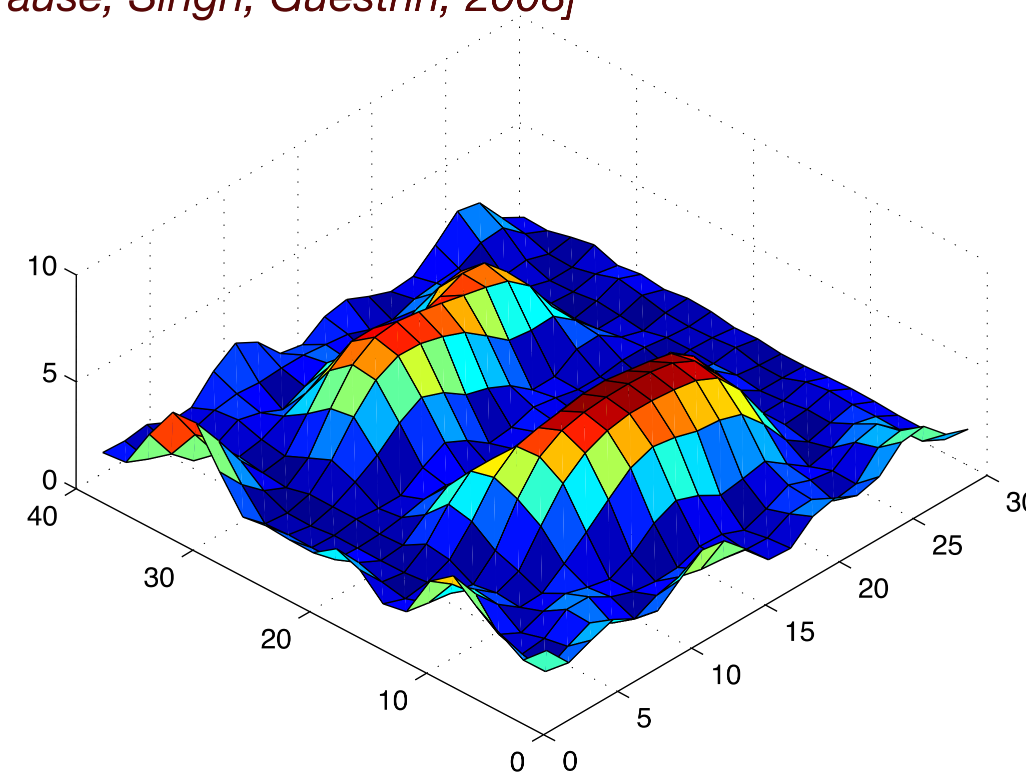
Modeling temperature distribution



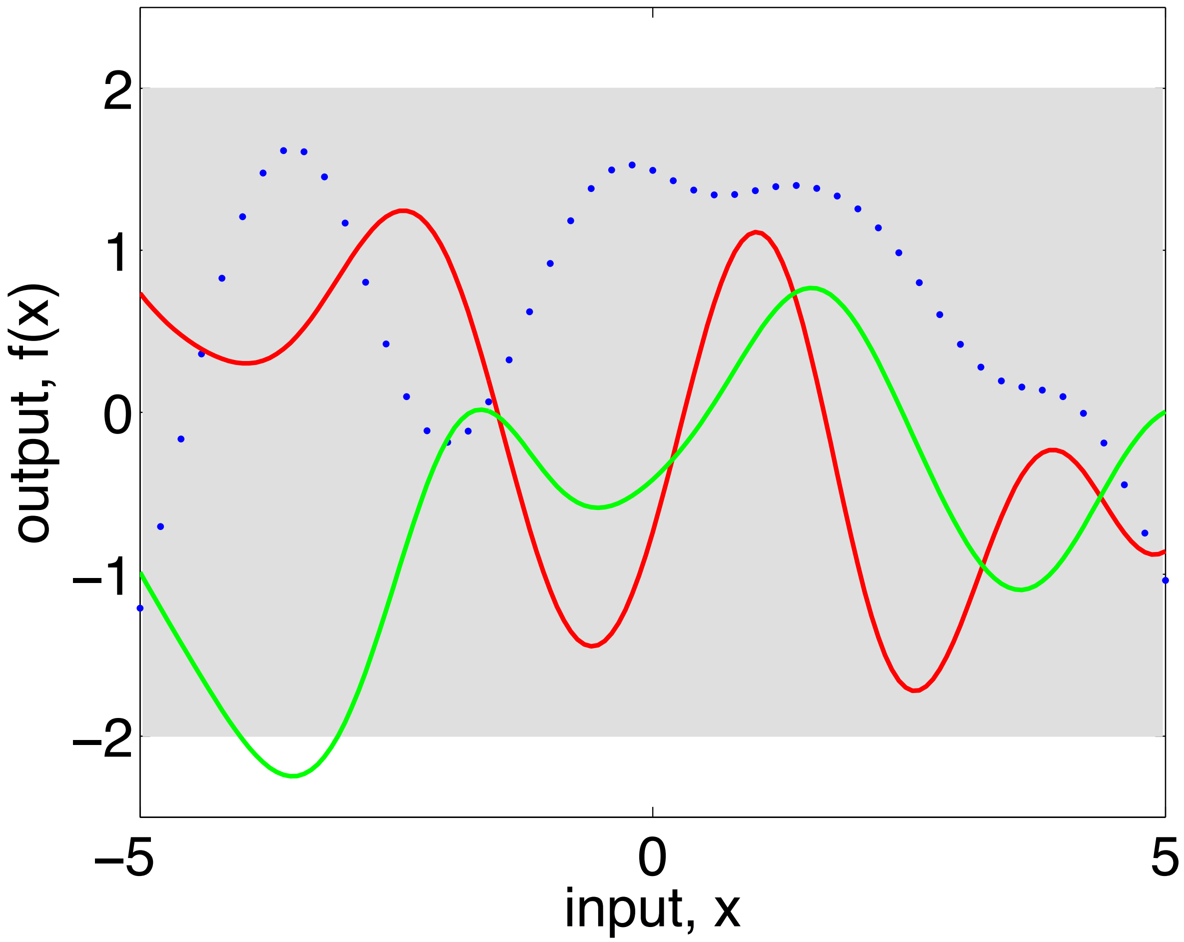
[Krause, Singh, Guestrin, 2008]



(a) Predicted temperature



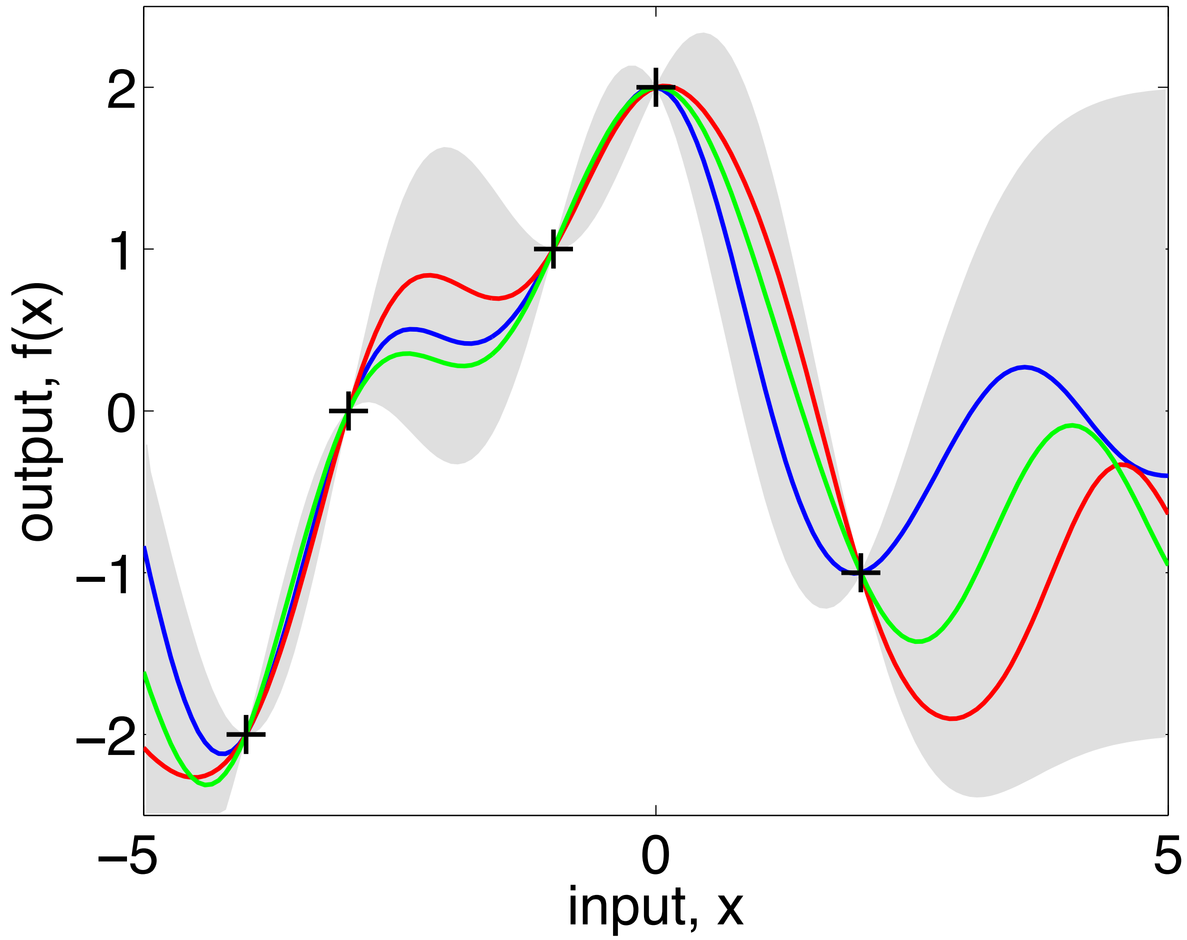
(a) Variance of predicted temperature



Dots: actual y values

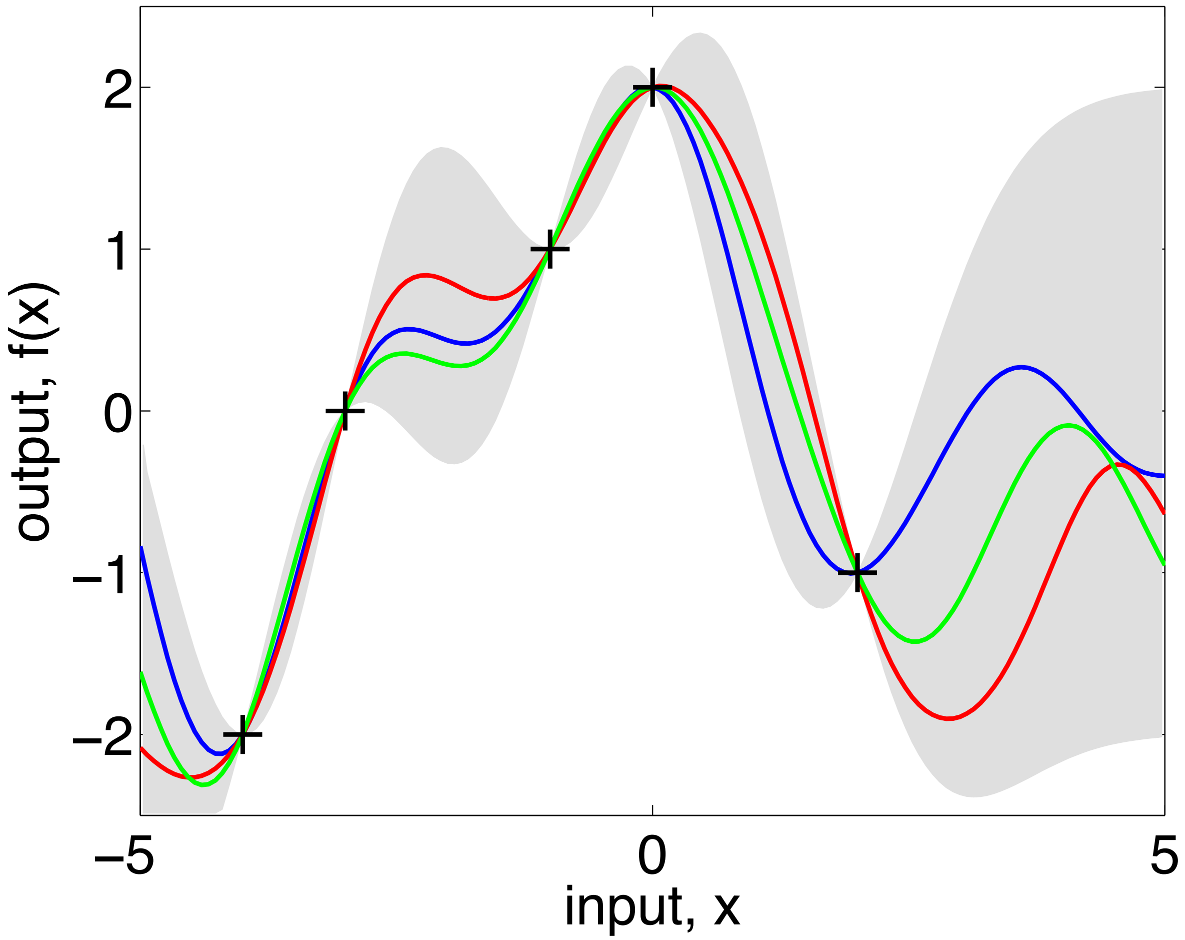
2 funcs sampled from GP using Gaussian-RBF kernel

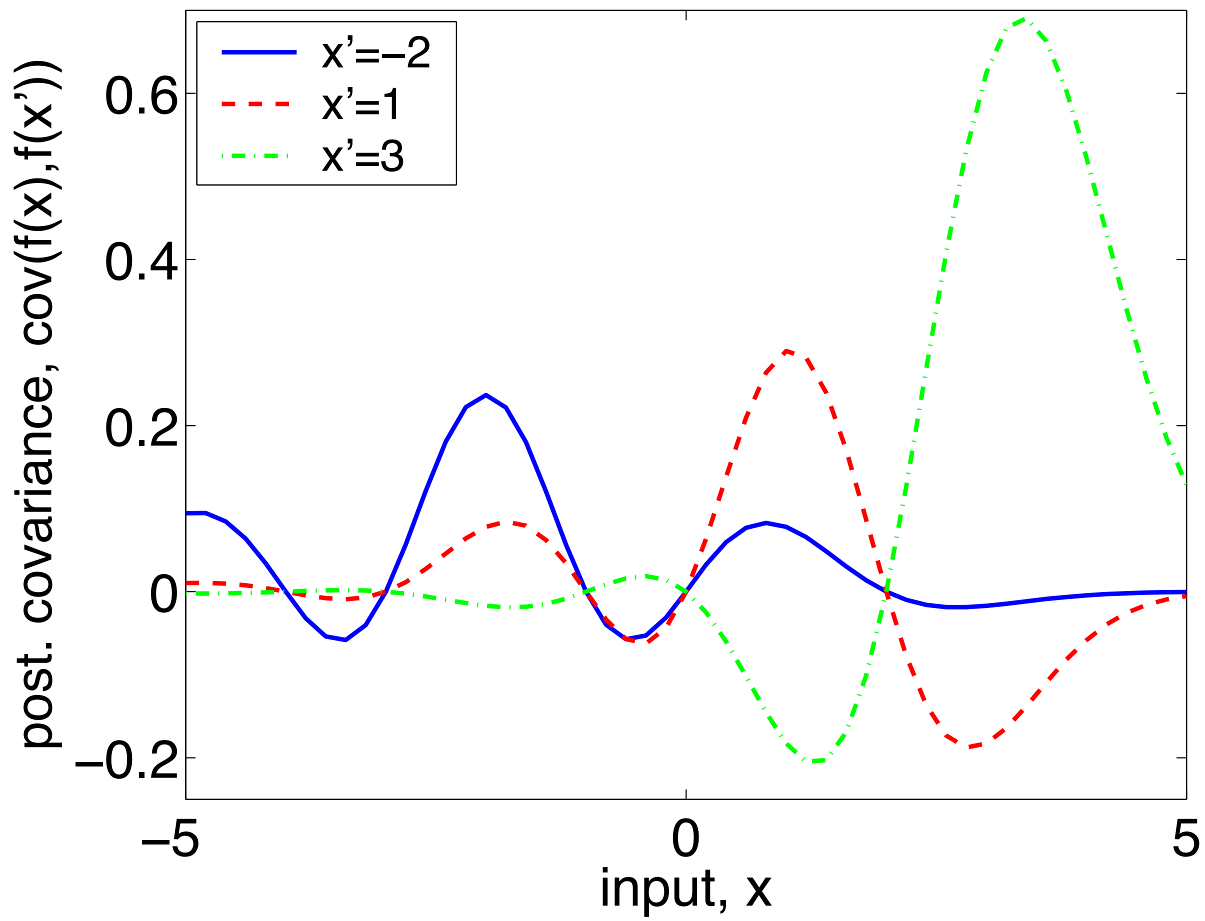
1. Obtain input / test points X_*
2. Compute covariance matrix $K(X_*, X_*)$
3. Generate random Gaussian vector



Posterior estimates
(i.e., prior conditioned on
5 noise free observations)

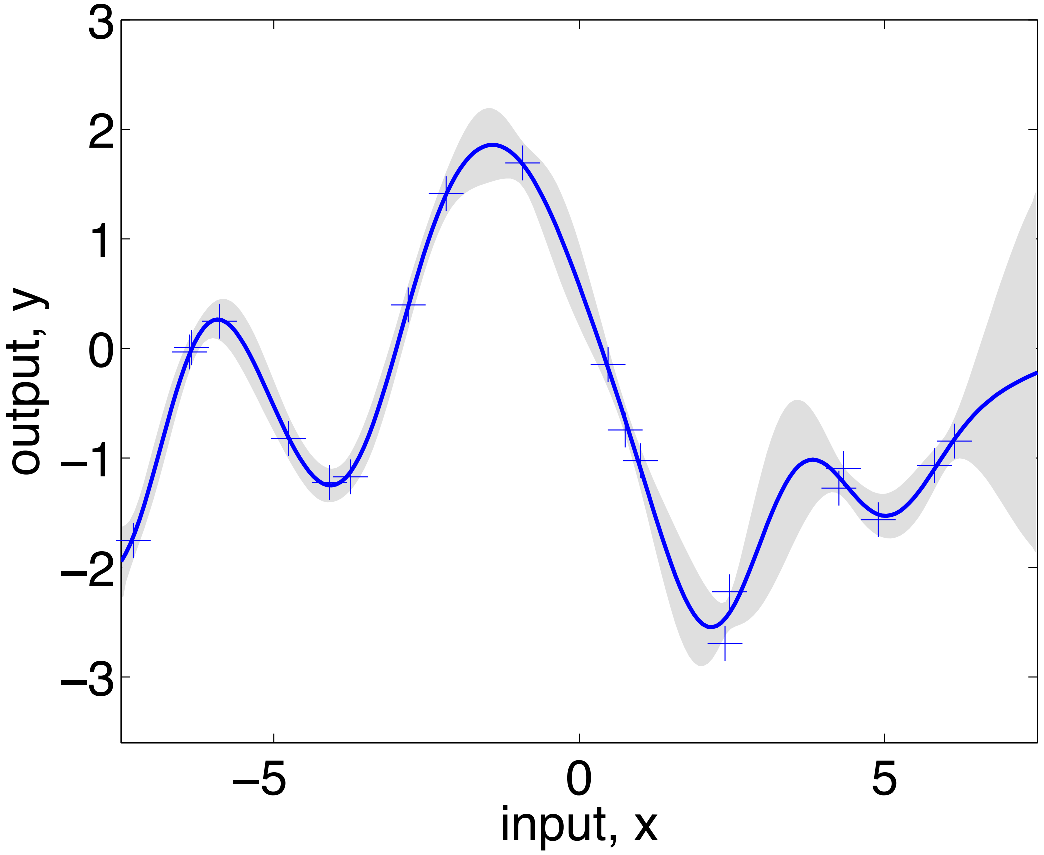
Gray area: 95% confidence intervals (pointwise mean ± 2 std-dev)



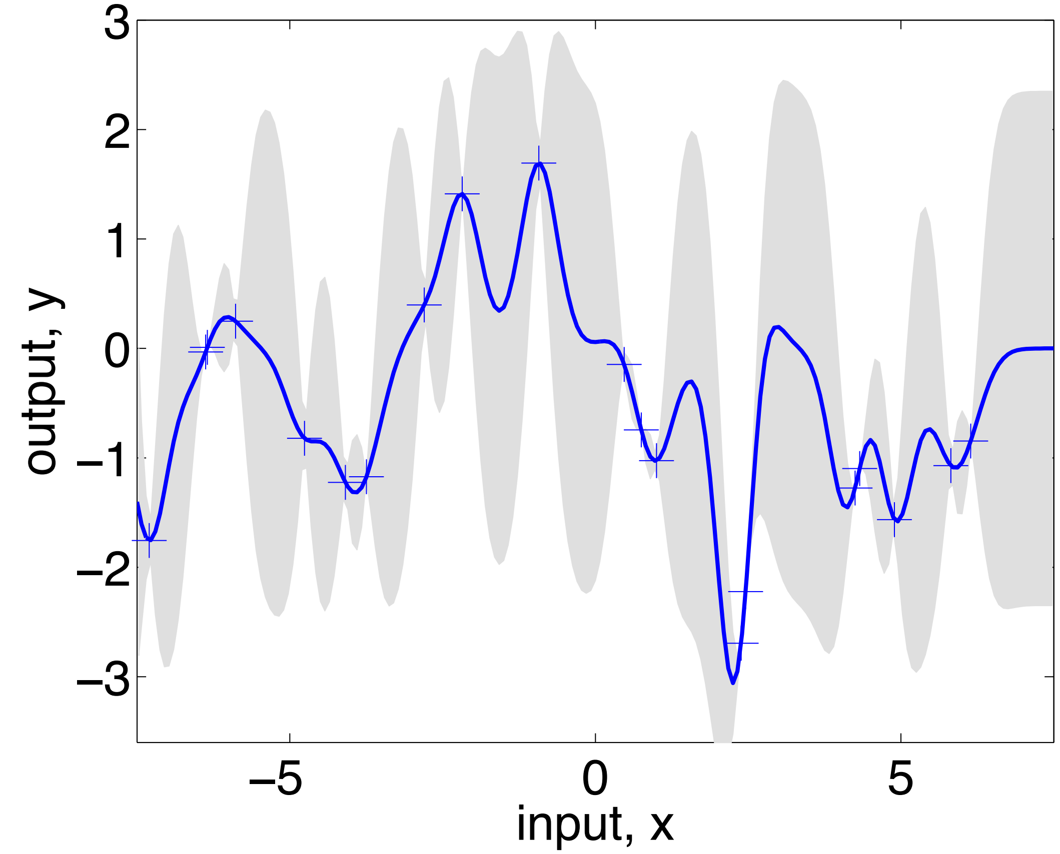


Observe: How posterior covariance between f(x) and f(x') depends on location covariance at close points is high and falls to zero at the training data

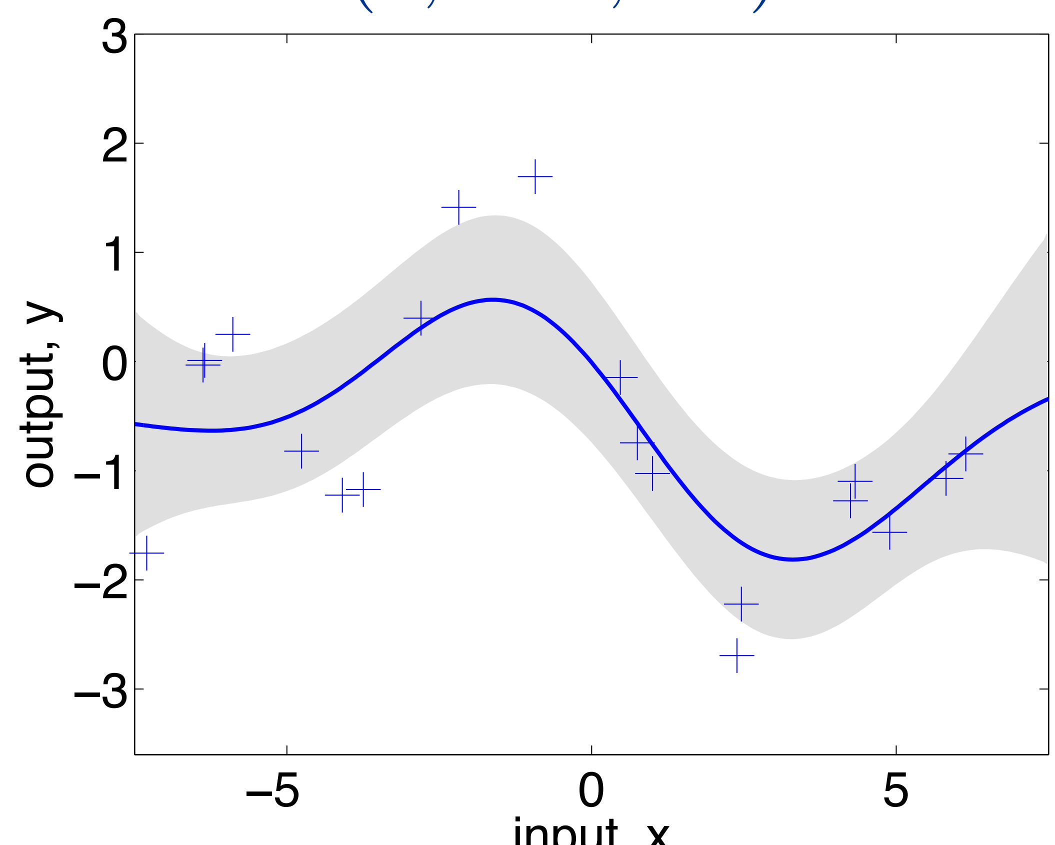
$(\ell, \sigma_f, \sigma_n) = (1, 1, 0.1)$



$(0.3, 1.08, 0.00005)$



$(3, 1.16, .89)$



$$k_y(x, x') = \sigma_f^2 \exp(-\frac{1}{2\ell^2}(x - x')^2) + \sigma_n^2 \delta_{xx'}$$

Useful links

http://www.gaussianprocess.org

http://ml.dcs.shef.ac.uk/gpss/gpws14/

https://github.com/SheffieldML/GPy