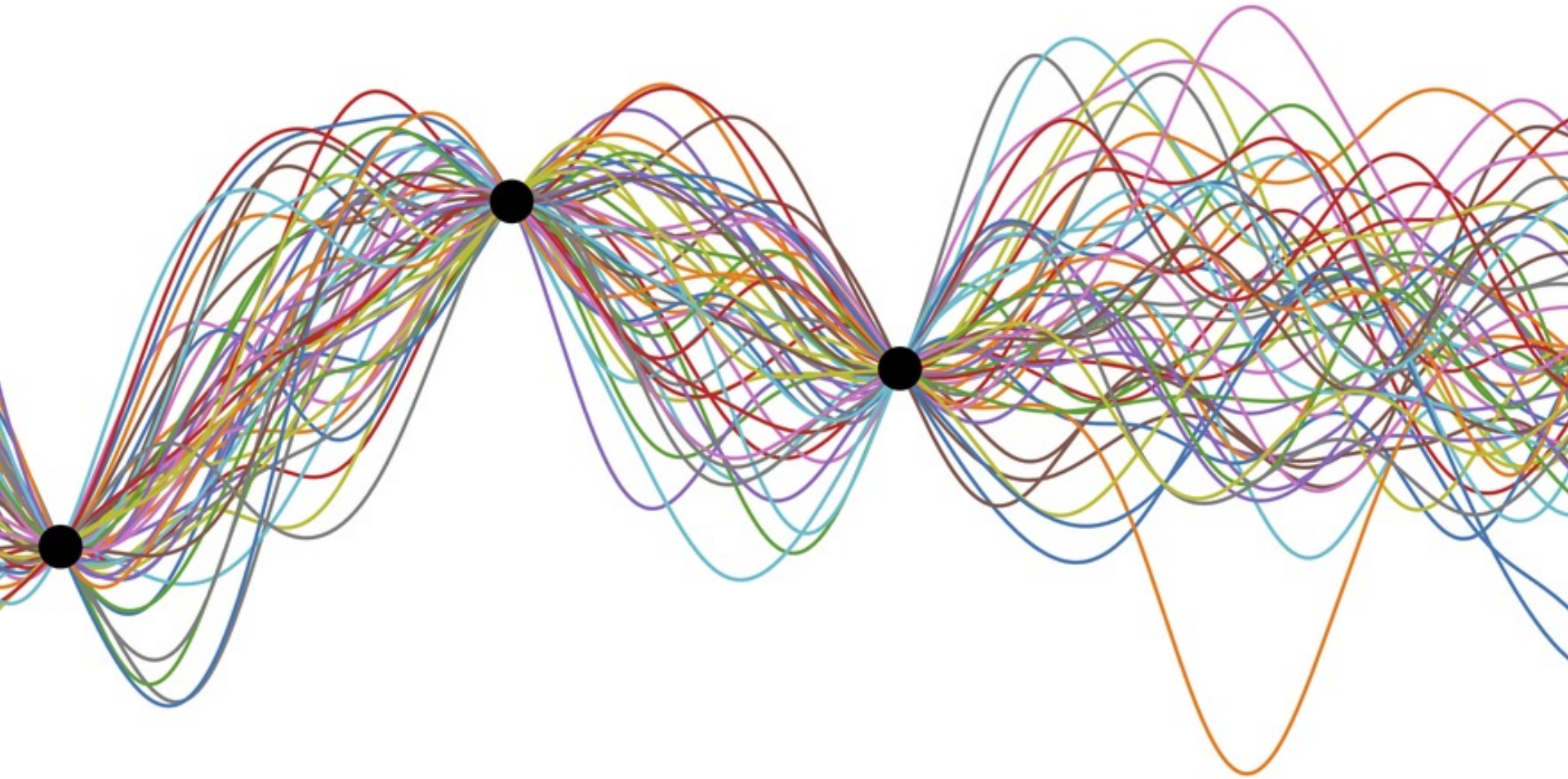
Lecture 23: Gaussian Processes



Goals for Today

Gaussian (normal) Distribution

- 1D & multivariate
- Conditional distribution (Bayes theorem)

Gaussian Processes for Regression

- Noise-free regression
- Regression with noise

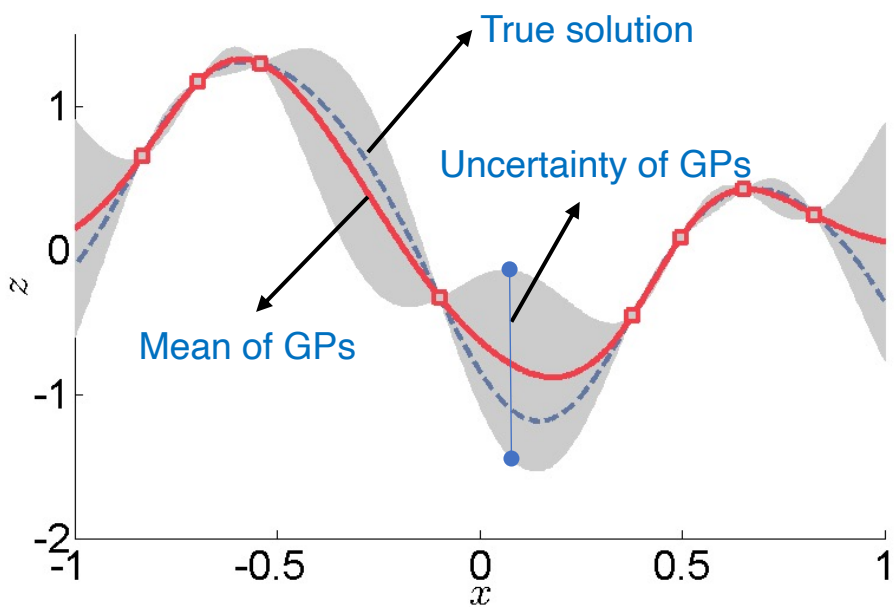
Gaussian Processes & Active Learning

Extended Topics on Gaussian Processes

- Kernel parameters estimation
- Relationship to other regression methods

Gaussian Processes - Introduction

- In supervised learning, given some inputs $\mathbf{x}_i$ and outputs $\mathbf{y}_i$, we assume $\mathbf{y}_i=\mathbf{f}(\mathbf{x}_i)$. The optimal approach is to infer a distribution over the functions given the data, which is called Gaussian processes (GPs) developed by Rasmussen (2004).
- A GP defines a prior over functions, which can be converted into a posterior over functions once we have seen some data. (Murphy 2012).
- A GP assumes that $\mathbf{p(f|X_i,y_i)}$ is a joint Gaussian with some mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\Sigma}$.



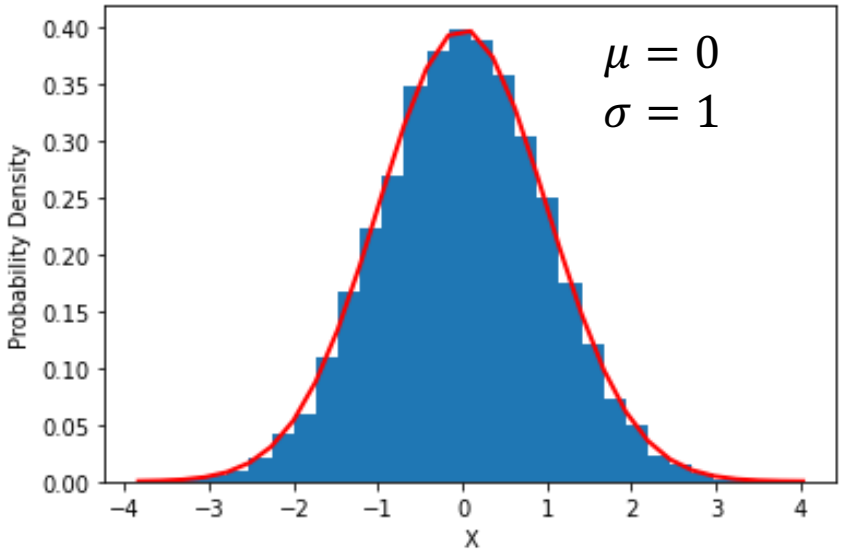
Gaussian Processes - Basics

- In probability theory, a Gaussian (or normal) distribution is a type of continuous probability distribution for a real-valued random variable. Conventionally, we write the Gaussian distribution of mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ as

$$\boldsymbol{f} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- In 1D Gaussian distribution, we have

$$f \sim \mathcal{N}(\mu, \Sigma = \sigma^2)$$



$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

$$\int_{-\infty}^{\infty} f(x)dx = 1$$

Gaussian Processes - Basics

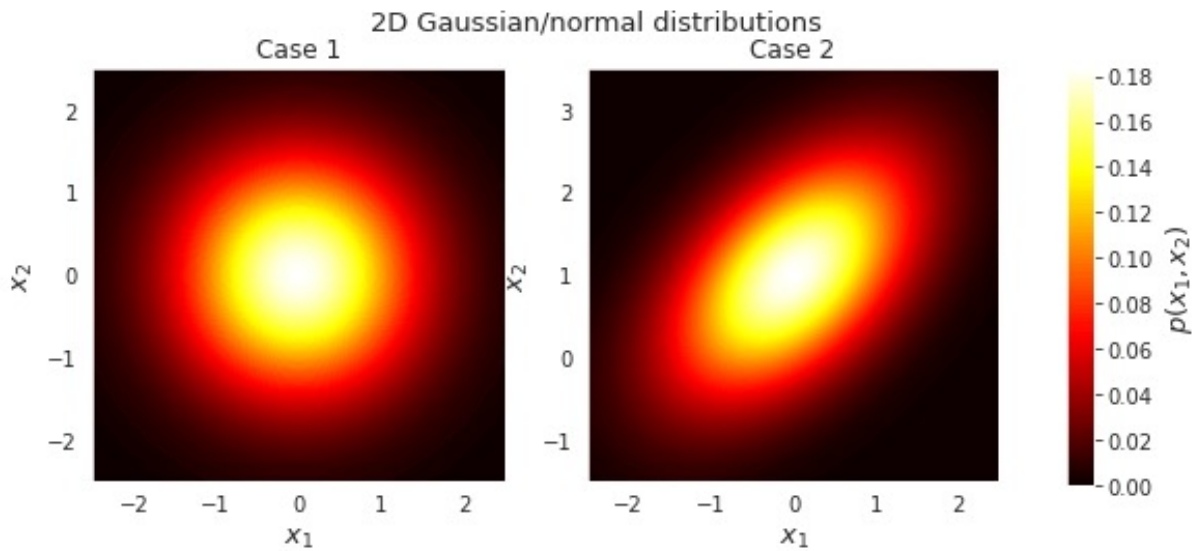
- For multivariate Gaussian distribution, the joint probability of dimension $\boldsymbol{d}$ is given by

$$p(\boldsymbol{x}\,|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d|\boldsymbol{\Sigma}|}}\exp\left(-\frac{1}{2}\underbrace{(\boldsymbol{x}-\boldsymbol{\mu})^T}_{d\times 1}\cdot\underbrace{\boldsymbol{\Sigma}^{-1}}_{d\times d}\cdot\underbrace{(\boldsymbol{x}-\boldsymbol{\mu})}_{d\times 1}\right)$$

Case 1:

$$\boldsymbol{\Sigma} = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$$

$$\boldsymbol{\mu} = \begin{bmatrix}0\\ 0\end{bmatrix}$$



Case 2:

$$\boldsymbol{\Sigma} = \begin{bmatrix}1 & 0.5\\ 0.5 & 1\end{bmatrix}$$

$$\boldsymbol{\mu} = \begin{bmatrix}0\\ 1\end{bmatrix}$$

Covariance $\boldsymbol{\Sigma}$ is a symmetric, positive definite matrix!

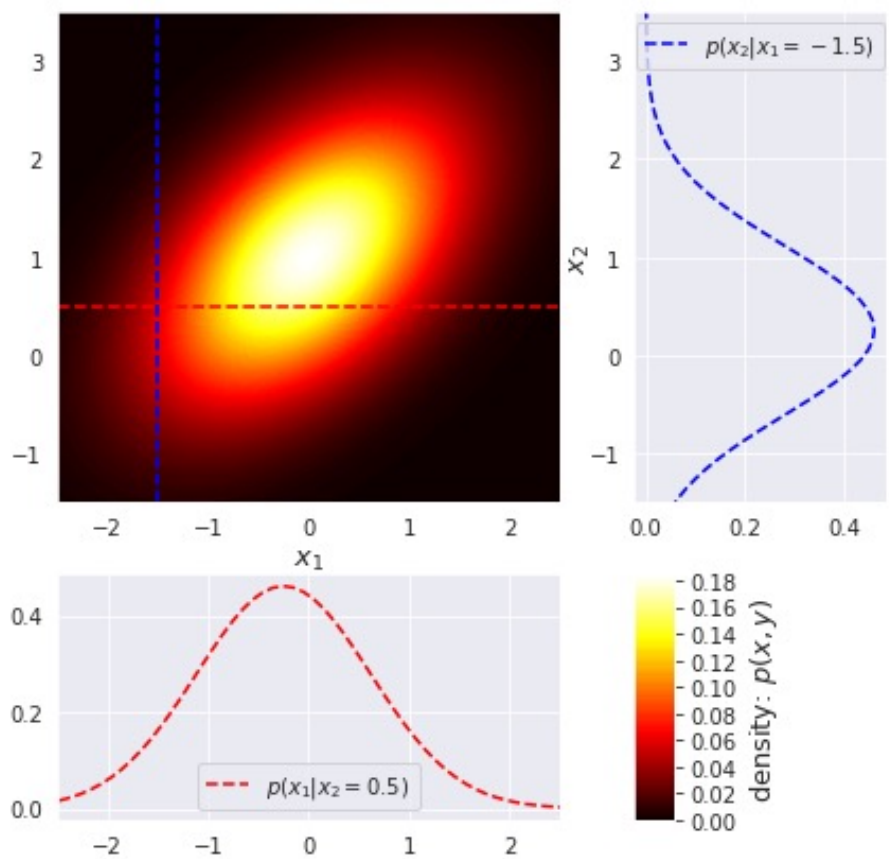
Gaussian Processes - Basics

- Conditional distribution of $\boldsymbol{x_1}$ given $\boldsymbol{x_2}$ is the probability distribution of $\boldsymbol{x_1}$ when $\boldsymbol{x_2}$ is known to be a particular value

Posterior distribution

$$p(\boldsymbol{x_1}\,|\boldsymbol{x_2}) = \frac{p(\boldsymbol{x_2}|\boldsymbol{x_1}) * p(\boldsymbol{x_1})}{p(\boldsymbol{x_2})}$$

Prior distribution



$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}\right)$$

$$p(x_1|x_2) = p_{1|2} = \mathcal{N}\left(\mu_{1|2}, \Sigma_{1|2}\right)$$

$$\mu_{1|2} = \mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(x_2 - \mu_2)$$

$$\Sigma_{1|2} = \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$$

Similarly, one can derive multidimensional conditional probability distribution (e.g. 2d)

$$p(x_1, x_2|x_3, x_4, \ldots) = \mathcal{N}\left(\mu_{1,2|3,4\ldots}, \Sigma_{1,2|3,4,\ldots}\right)$$

Gaussian Processes for Regression

- Now we discuss GPs for regression. Let the prior on the regression function be a Gaussian process such that

$$f(\boldsymbol{x}) \sim GP\big(m(\boldsymbol{x}), \kappa(\boldsymbol{x}, \boldsymbol{x})\big) = N(f|\boldsymbol{\mu}, \boldsymbol{K})$$

<u>Mean **function**</u>:

$$m(\boldsymbol{x}) = E[f(\boldsymbol{x})], \boldsymbol{\mu} = (m(\boldsymbol{x}_1), \dots, m(\boldsymbol{x}_N))$$

<u>Covariance **function** kernel</u>:

$$\kappa(\boldsymbol{x}, \boldsymbol{x}) = E[(f(\boldsymbol{x}) - m(\boldsymbol{x}))(f(\boldsymbol{x}) - m(\boldsymbol{x}))]$$

$$K_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$$

- It is common to use a mean **function** of $m(\boldsymbol{x}) = 0$ for the prior.
- The kernel selection highly depend on the considered problem. Typical choice is a **squared exponential kernel** (Gaussian kernel or RBF kernel)

$$\kappa(\boldsymbol{x_i}, \boldsymbol{x_j}) = \sigma_f^2 \exp\left(-\frac{1}{2l^2}\|x_i - x_j\|_2^2\right)$$

$L2$ norm

$\sigma_f = 1$ is the vertical variance

l is the tuning parameter

Gaussian Processes – Noise-free Regression

- Given a training data set $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{f}_i), i = 1, \dots, N\}$, where $\boldsymbol{f}_i$ is the noise-free observation (output function), we want to predict $\boldsymbol{f}_*$ for a test set $\boldsymbol{x}_*$

- By the definition of GPs, the joint distribution has following distribution

$$\begin{pmatrix} \boldsymbol{f} \\ \boldsymbol{f}_* \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu}_* \end{pmatrix}, \begin{pmatrix} \boldsymbol{K} & \boldsymbol{K}_* \\ \boldsymbol{K}_* & \boldsymbol{K}_{**} \end{pmatrix} \right)$$

Prior distribution

$$p(\boldsymbol{f}_* | \boldsymbol{x}_*, \boldsymbol{x}, \boldsymbol{f}) = \mathcal{N}(\boldsymbol{f}_* | \boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$$

$$\boldsymbol{\mu}_* = \boldsymbol{\mu}(\boldsymbol{X}_*) + \boldsymbol{K}_*^T \boldsymbol{K}^{-1} (\boldsymbol{f} - \boldsymbol{\mu}(\boldsymbol{x}))$$

Posterior distribution

$$\boldsymbol{\Sigma}_* = \boldsymbol{K}_{**} - \boldsymbol{K}_*^T \boldsymbol{K}^{-1} \boldsymbol{K}_*$$

$$\boldsymbol{K}_{**} = \kappa(\boldsymbol{x}_*, \boldsymbol{x}_*) \qquad \boldsymbol{K}_* = \kappa(\boldsymbol{x}, \boldsymbol{x}_*) \qquad \boldsymbol{K} = \kappa(\boldsymbol{x}, \boldsymbol{x})$$

<u>Typically, we let μ=0; then</u> $\boldsymbol{\mu}_* = \boldsymbol{K}_*^T \boldsymbol{K}^{-1} \boldsymbol{f}$

Gaussian Processes – Implementation

- GPs is a rather computationally expensive method; the major computation time is consumed for the inversion of $\boldsymbol{K}$, which is about $\boldsymbol{O(N^3)}$.

- In addition, a simple inversion operation $\boldsymbol{K^{-1}}$ is unstable. There are two solutions:

1. First compute the Cholesky decomposition, $\boldsymbol{K = LL^T}$; next, $\boldsymbol{K^{-1}y = L^{-T}L^{-1}y}$. ($\boldsymbol{L}$ is lower triangular matrix)

Algorithm 15.1: GP regression

$\mathbf{L} = \text{cholesky}(\mathbf{K} + \sigma_y^2 \mathbf{I})$;
$\boldsymbol{\alpha} = \mathbf{L}^T \backslash (\mathbf{L} \backslash \mathbf{y})$;
$\mathbb{E}[f_*] = \mathbf{k}_*^T \boldsymbol{\alpha}$;
$\mathbf{v} = \mathbf{L} \backslash \mathbf{k}_*$;
$\text{var}[f_*] = \kappa(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{v}^T \mathbf{v}$;
$\log p(\mathbf{y}|\mathbf{X}) = -\frac{1}{2}\mathbf{y}^T \boldsymbol{\alpha} - \sum_i \log L_{ii} - \frac{N}{2}\log(2\pi)$

2. Another approach is to solve the linear equation, namely $\boldsymbol{K\alpha = y}$ (e.g. conjugate gradient)

Gaussian Processes – Implementation

```
K = kernel(X, X) # compute the kernel function with training data X
L = np.linalg.cholesky(K) # Cholesky decomposition
Lk = np.linalg.solve(L, kernel(X, Xtest)) # Inv(L)*kernel(X, Xtest)

# compute the mean at our test points.
mu = np.dot(Lk.T, np.linalg.solve(L, y)) # y is training outcome
# compute the variance at our test points.
K_ = kernel(Xtest, Xtest)
s2 = np.diag(K_) - np.sum(Lk**2, axis=0) # variance
s = np.sqrt(s2) # stardard deviation
```

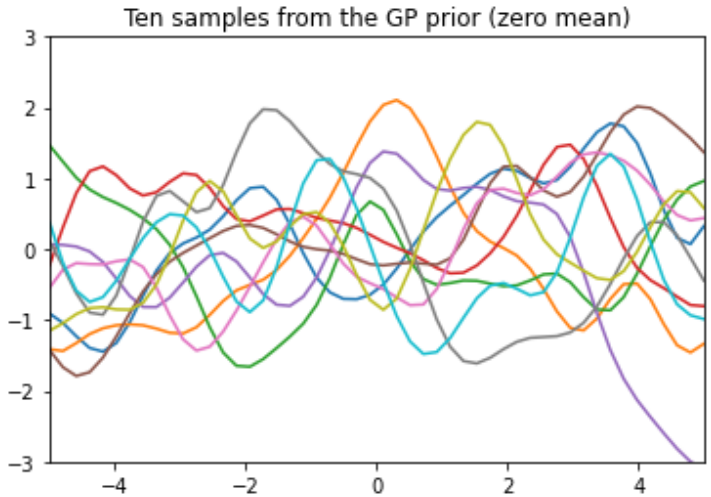
(1) Using Cholesky decomposition to compute $\boldsymbol{L}$ for $\boldsymbol{K} = \boldsymbol{L}\boldsymbol{L}^{\boldsymbol{T}}$; it is much easier (more stable) for invert $\boldsymbol{L}$ than $\boldsymbol{K}$

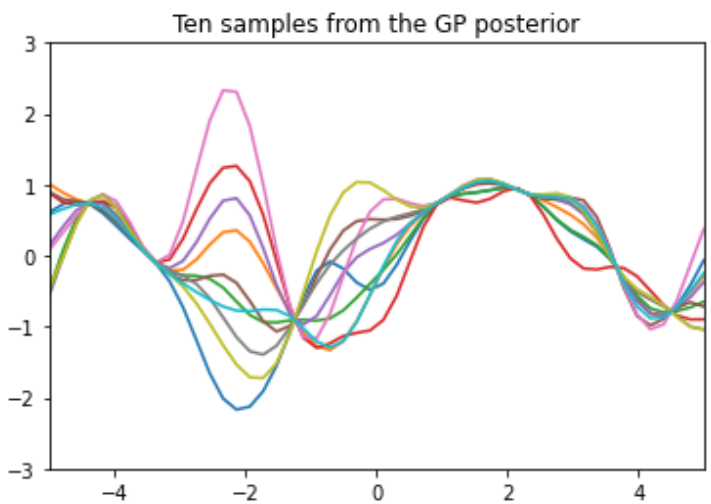
(2) 'mu' is the mean function for test data point, which is the curve we try to predict;

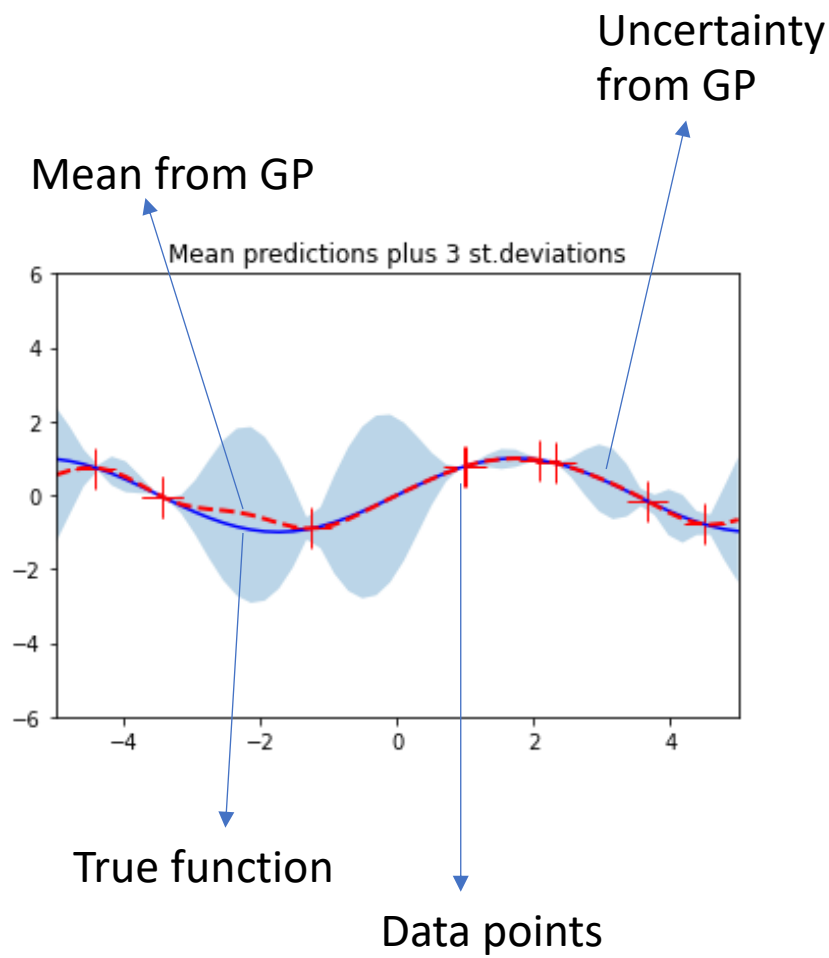
(3) 's' is the standard deviation for the uncertainty of the predicted curve. <u>The power of GPs comes from the ability to predict the reliable estimate of uncertainty!</u>

Gaussian Processes – Example (noise-free)

- We assume the prior to be zero mean ($\mu = 0$) and kernel function to be RBF ($l = 0.5$).

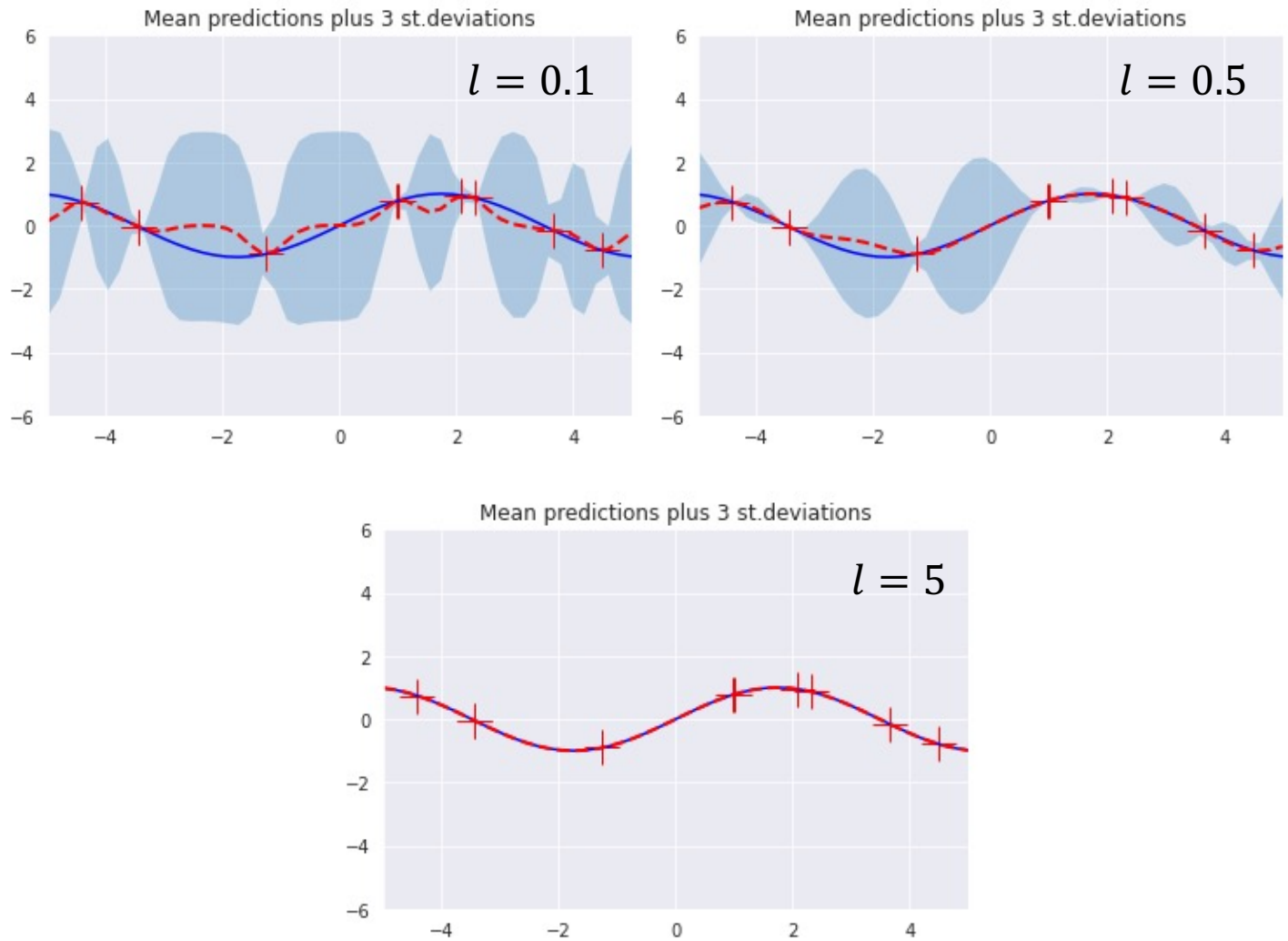






Gaussian Processes – Kernel Parameter

- We examine how kernel parameter affect the mean prediction as well as uncertainty.



Gaussian Processes – Regression with Noise

- Now let us consider the case where what we observe is a noisy function, $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\sigma}_y^2)$.
- The covariance of the noisy data is $\mathrm{cov}[\boldsymbol{y}|\boldsymbol{X}] = \boldsymbol{K} + \sigma_y^2 \boldsymbol{I} \equiv \boldsymbol{K}_y$.
- The joint density of the observed data and test points ($\boldsymbol{X}_*$); again, we typically assume the mean of the prior to be zero ($\boldsymbol{\mu} = \boldsymbol{0}$)

$$\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu}_* \end{pmatrix}, \begin{pmatrix} \boldsymbol{K}_y & \boldsymbol{K}_* \\ \boldsymbol{K}_* & \boldsymbol{K}_{**} \end{pmatrix} \right)$$

$$p(\boldsymbol{f}_* | \boldsymbol{X}_*, \boldsymbol{X}, \boldsymbol{y}) = \mathcal{N}(\boldsymbol{f}_* | \boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$$

$$\boldsymbol{\mu}_* = \boldsymbol{\mu}(\boldsymbol{X}_*) + \boldsymbol{K}_*^T \boldsymbol{K}_y^{-1} (\boldsymbol{f} - \boldsymbol{\mu}(\boldsymbol{X}))$$

$$\boldsymbol{\Sigma}_* = \boldsymbol{K}_{**} - \boldsymbol{K}_*^T \boldsymbol{K}_y^{-1} \boldsymbol{K}_*$$

$\boldsymbol{\mu} = 0$ →

$$\begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{f}_* \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{0} \\ \boldsymbol{\mu}_* \end{pmatrix}, \begin{pmatrix} \boldsymbol{K}_y & \boldsymbol{K}_* \\ \boldsymbol{K}_* & \boldsymbol{K}_{**} \end{pmatrix} \right)$$

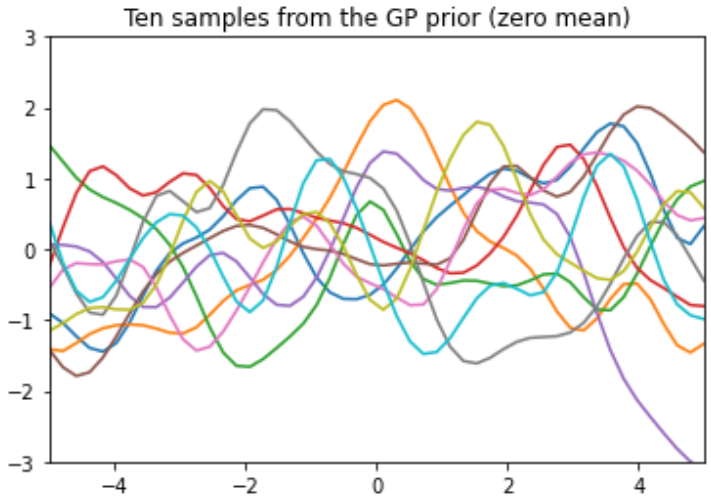
$$p(\boldsymbol{f}_* | \boldsymbol{X}_*, \boldsymbol{X}, \boldsymbol{y}) = \mathcal{N}(\boldsymbol{f}_* | \boldsymbol{\mu}_*, \boldsymbol{\Sigma}_*)$$

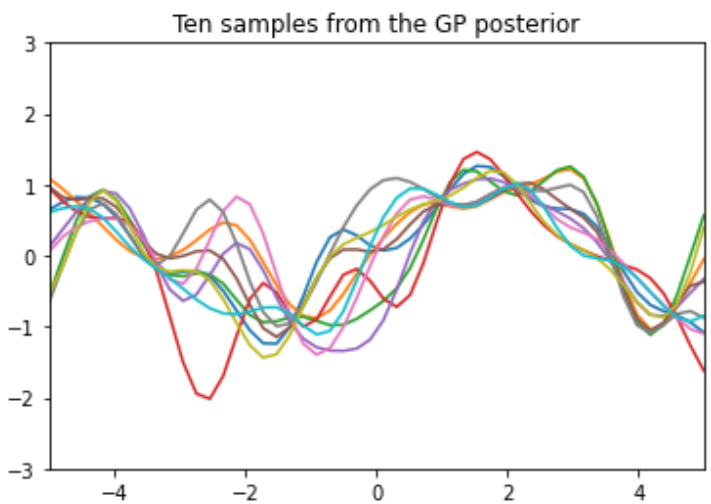
$$\boldsymbol{\mu}_* = \boldsymbol{K}_*^T \boldsymbol{K}_y^{-1} \boldsymbol{y}$$

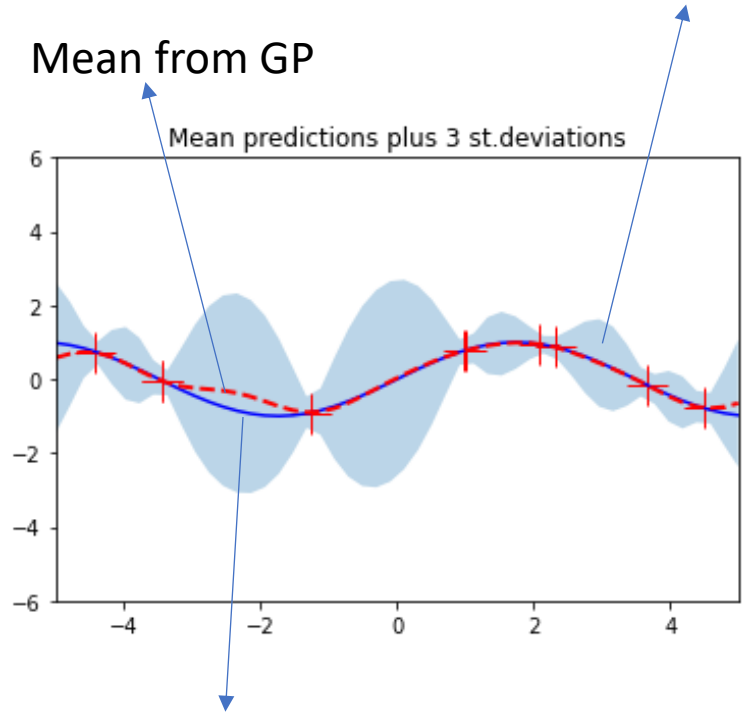
$$\boldsymbol{\Sigma}_* = \boldsymbol{K}_{**} - \boldsymbol{K}_*^T \boldsymbol{K}_y^{-1} \boldsymbol{K}_*$$

Gaussian Processes – Example (with noise)

- We assume the prior to be zero mean ($\boldsymbol{\mu} = 0$) and kernel function to be RBF ($l = 0.5, \sigma_y = 0.01$).







Gaussian Processes – Kernel Parameters Estimation

- To estimate the kernel parameters, we could exhaust the search of a discrete values with validation as an objective. However, this can be very slow.

- The marginal likelihood is $p(\boldsymbol{f}|\boldsymbol{X}) = \mathcal{N}(\boldsymbol{f}|\boldsymbol{0}, \boldsymbol{K})$

$$\log p(\boldsymbol{y}|\boldsymbol{X}) = \log \mathcal{N}(\boldsymbol{y}|\boldsymbol{0}, \boldsymbol{K}_{\boldsymbol{y}}) = -\frac{1}{2}\boldsymbol{y}^{\boldsymbol{T}}\boldsymbol{K}_{\boldsymbol{y}}^{-\boldsymbol{1}}\boldsymbol{y} - \frac{1}{2}\log\left|\boldsymbol{K}_{\boldsymbol{y}}\right| - \frac{N}{2}\log(2\pi)$$

To <u>maximize</u> the likelihood, let the kernel be the hyper-parameters denoted as $\theta_i = (l, \sigma_y)$ and then apply gradient based approach

$$\frac{\partial}{\partial \theta_i}\log p(\boldsymbol{y}|\boldsymbol{X}) = \frac{1}{2}\boldsymbol{y}^{\boldsymbol{T}}\frac{\partial \boldsymbol{K}_{\boldsymbol{y}}^{-\boldsymbol{1}}}{\partial \theta_i}\boldsymbol{y} - \frac{1}{2}\mathrm{tr}\left(\boldsymbol{K}_{\boldsymbol{y}}^{-\boldsymbol{1}}\frac{\partial \boldsymbol{K}_{\boldsymbol{y}}}{\partial \theta_i}\right)$$

- In case, the above method is not convex, other optional methods include:
  (1) Optimize the posterior probability
  (2) Multiple kernel learning (optimize weight instead of kernel parameters)

Gaussian Processes – Connection to Other Methods

- Connecting back to Ridge Regression:

$$\boldsymbol{\beta} = \left(\boldsymbol{X}^T\boldsymbol{X} + \boldsymbol{\sigma}^2\boldsymbol{I}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\boldsymbol{\beta} = \boldsymbol{X}^T\boldsymbol{X}\left(\boldsymbol{X}^T\boldsymbol{X} + \boldsymbol{\sigma}^2\boldsymbol{I}\right)^{-1}\boldsymbol{y}$$

Let $\boldsymbol{K} = \boldsymbol{X}^T\boldsymbol{X}$, we have $\hat{\boldsymbol{y}} = \boldsymbol{K}\left(\boldsymbol{K} + \boldsymbol{\sigma}^2\boldsymbol{I}\right)^{-1}\boldsymbol{y}$, which is exactly the posterior form of GPs.

- Infinitely large Neural Nets (one hidden layer) → GPs
  - Output function: $z^1(\boldsymbol{x}) = \beta_0^1 + \sum_{j=1}^N x_j^1(\boldsymbol{x})\beta_j^1$
  - $x_j^1(\boldsymbol{x}) = \phi\left(\beta_j^0 + \sum_{k=1}^N x_k\beta_{jk}^0\right)$ is the activation function
  - Assume that β_0^1 and β_j^1 are independent and normal distribution variable
  - Central limit theorem:

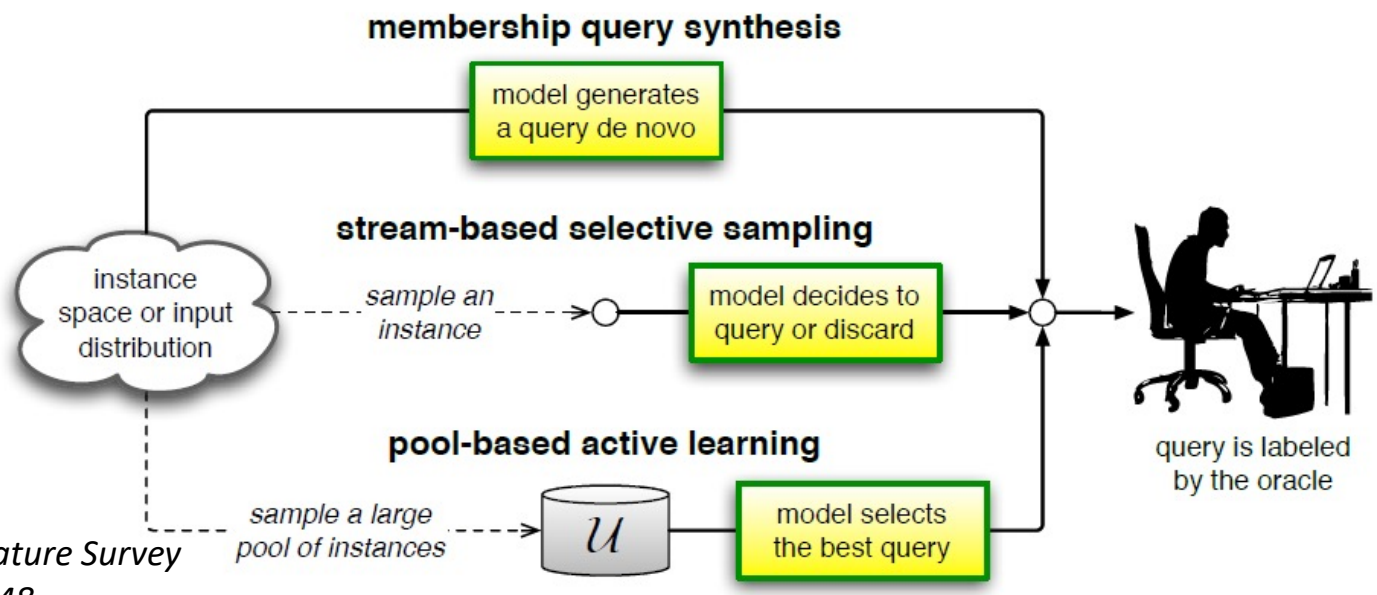
$$\lim_{N\to\infty}\sum_{j=1}^N x_j^1(\boldsymbol{x})\beta_j^1 = \lim_{N\to\infty}\frac{1}{N}\sum_{j=1}^N x_j^1(\boldsymbol{x})\tilde{\beta}_j^1 \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{K})$$

Active Learning

- Active learning is a machine learning algorithm can achieve greater accuracy with fewer labeled training instances if it is allowed to choose the data from which is learns.

- An active learner may ask queries in the form of unlabeled instances to be labeled by an oracle (e.g., a human annotator or a machine)

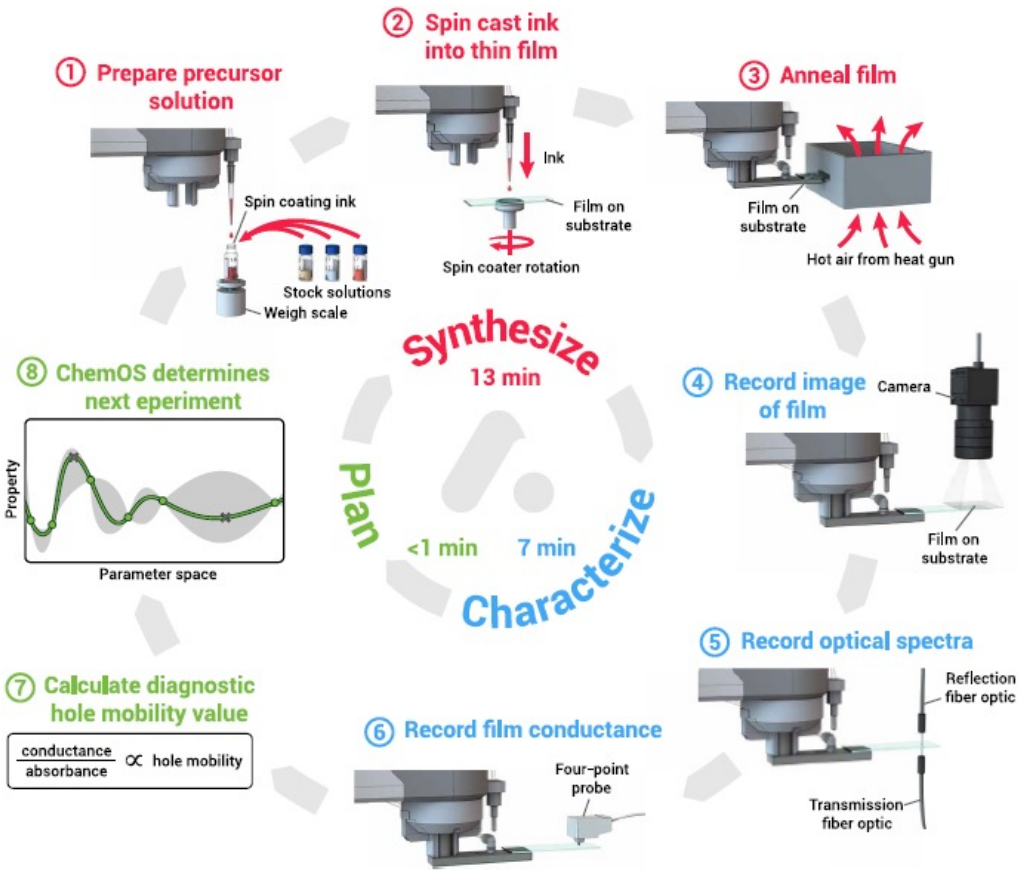
- Active learning is particularly suitable when <u>unlabeled data may be abundant but labels are difficult, time-consuming, or expensive to obtain</u>



Burr Settles (2009)
Active Learning Literature Survey
Technical Report #1648

Active Learning - Gaussian Processes

- Because GPs offer the uncertainty of the prediction, the algorithm can provide an effective query mechanism for active learning.
- A recent interesting example: <u>Self-driving laboratory for accelerated discovery of thin-film materials</u>



- Discovering and optimizing commercially viable materials (e.g. for clean energy applications) typically takes more than a decade

- Material synthesize and characterization offer the data point for planning

- Active learning combining with GPs completes the query, thereof entire process is automatic.

MacLeod et al., Sci. Adv. 2020; 6 : eaaz8867