ENM 360: Introduction to Data-driven Modeling

Lecture #9: Bayesian linear regression

Paris Perdikaris
September 29, 2019

Penn
University of Pennsylvania

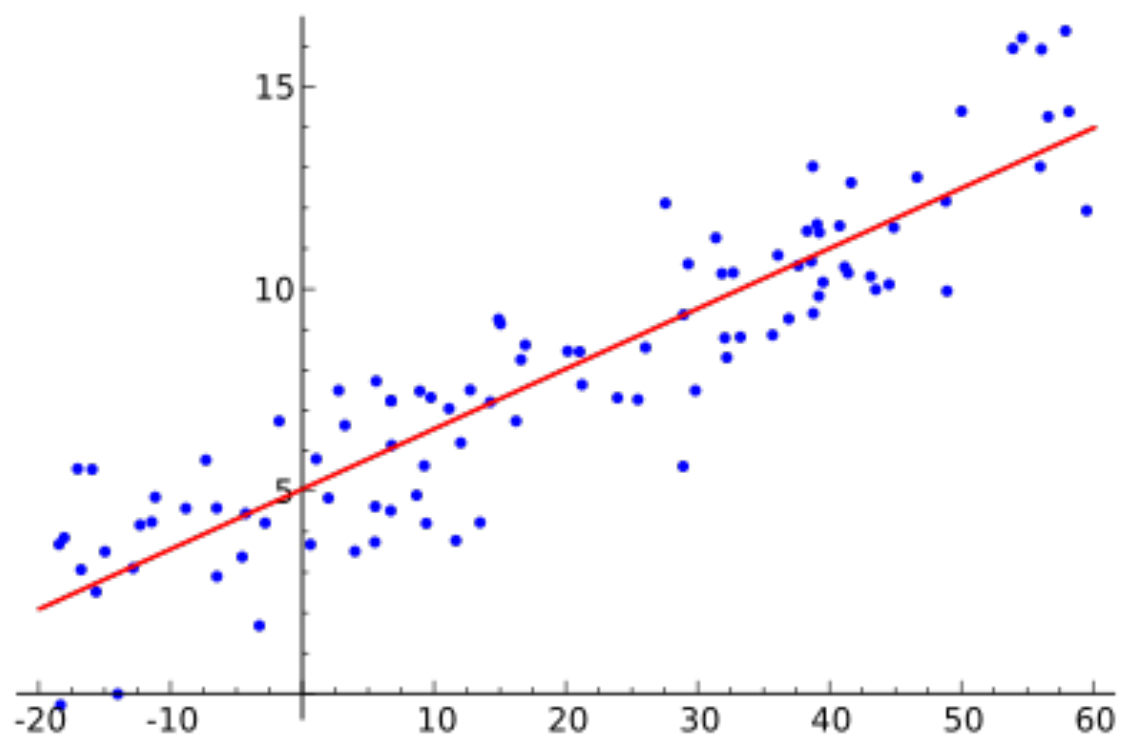
Linear regression

$$f : \mathcal{X} \to \mathcal{Y}$$

$$\mathcal{D} = \{\boldsymbol{x}, \boldsymbol{y}\},\ \boldsymbol{x} \in \mathcal{X},\ \boldsymbol{y} \in \mathcal{Y}$$

$$\boldsymbol{y} = f(\boldsymbol{x}) + \epsilon$$

$$f(\boldsymbol{x}) = w^T \boldsymbol{x}$$



"It's <u>not</u> just about lines and planes!"

Linear regression with basis functions

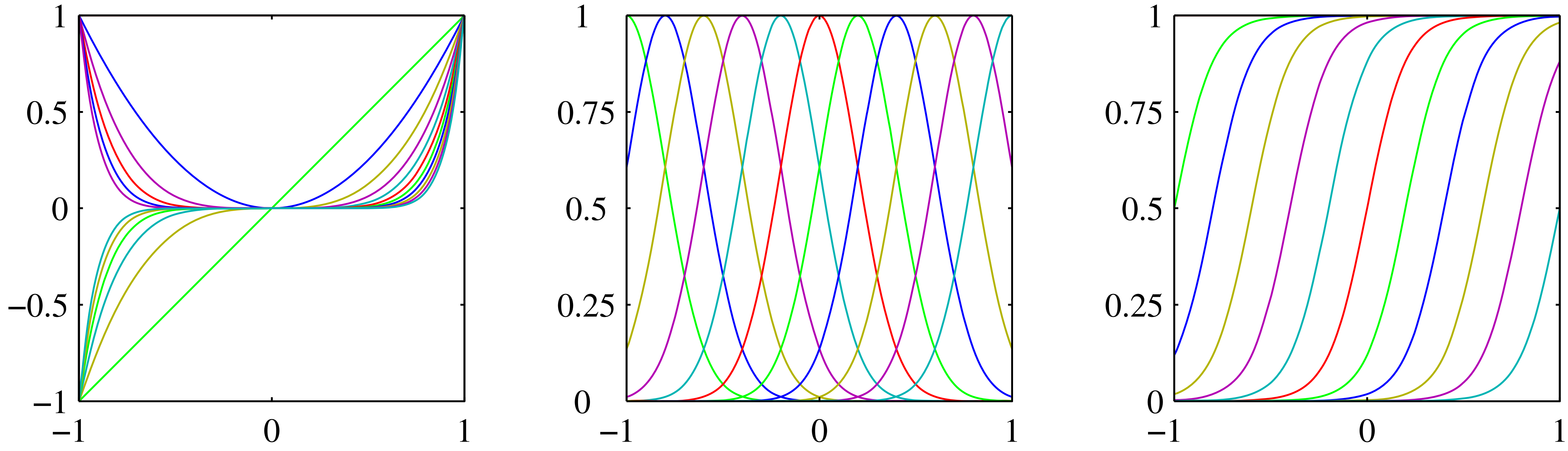
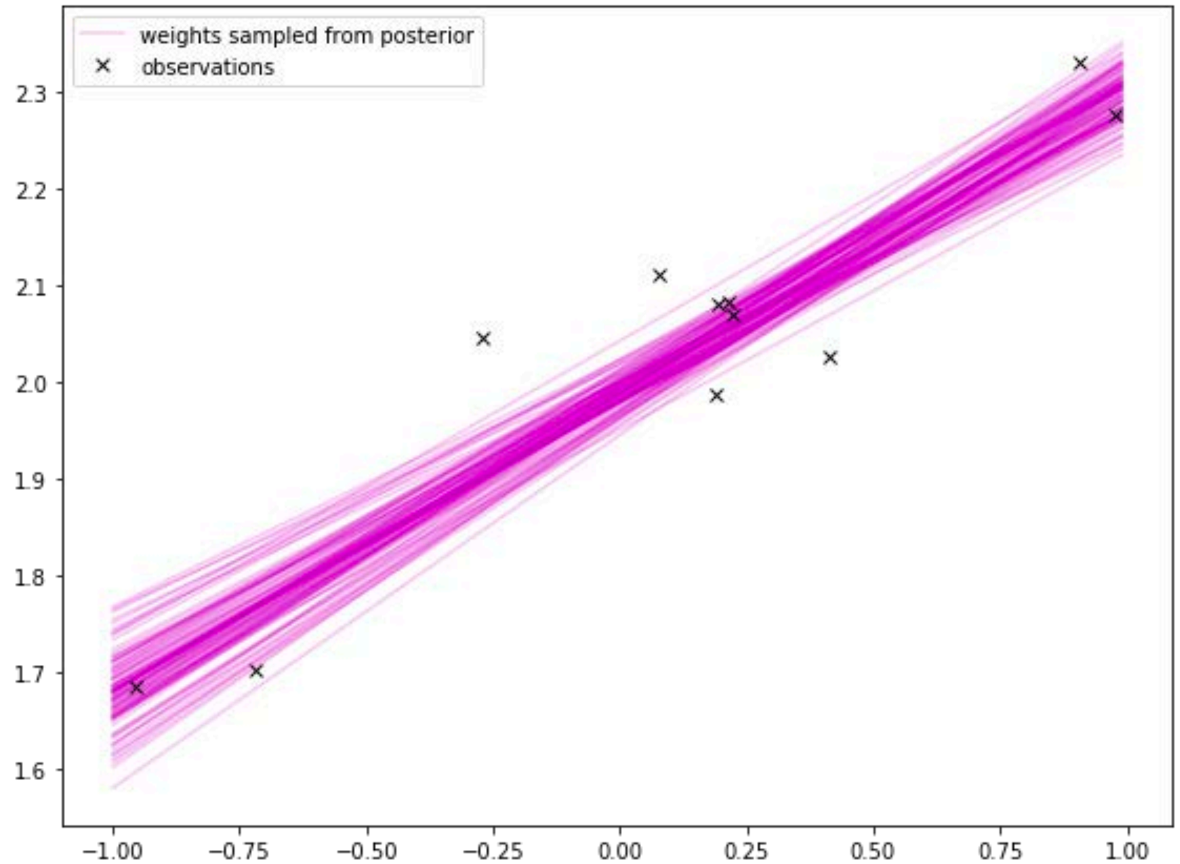
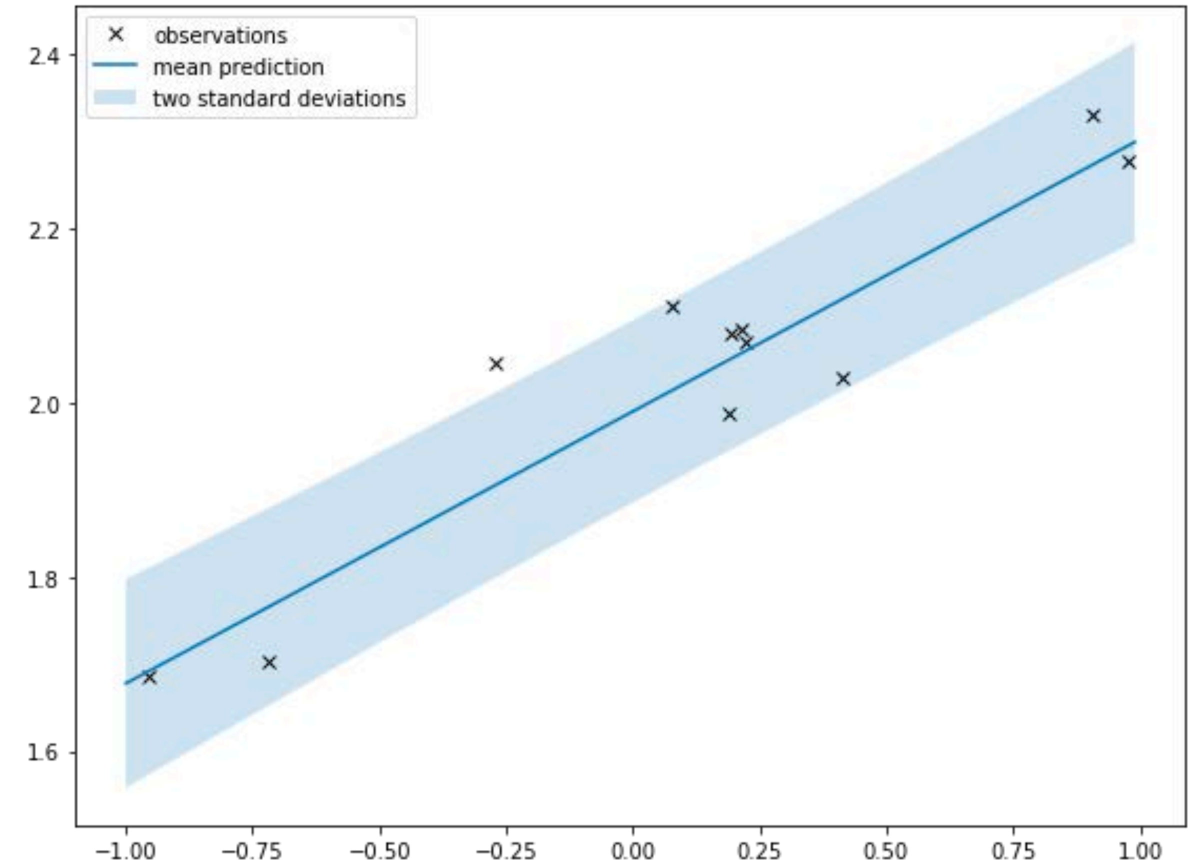
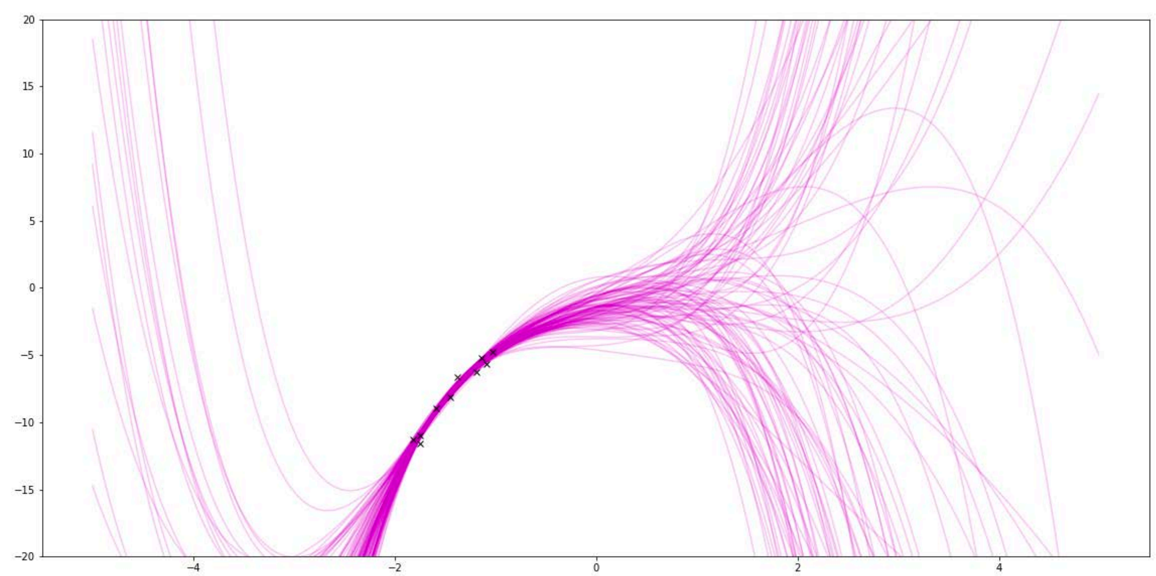


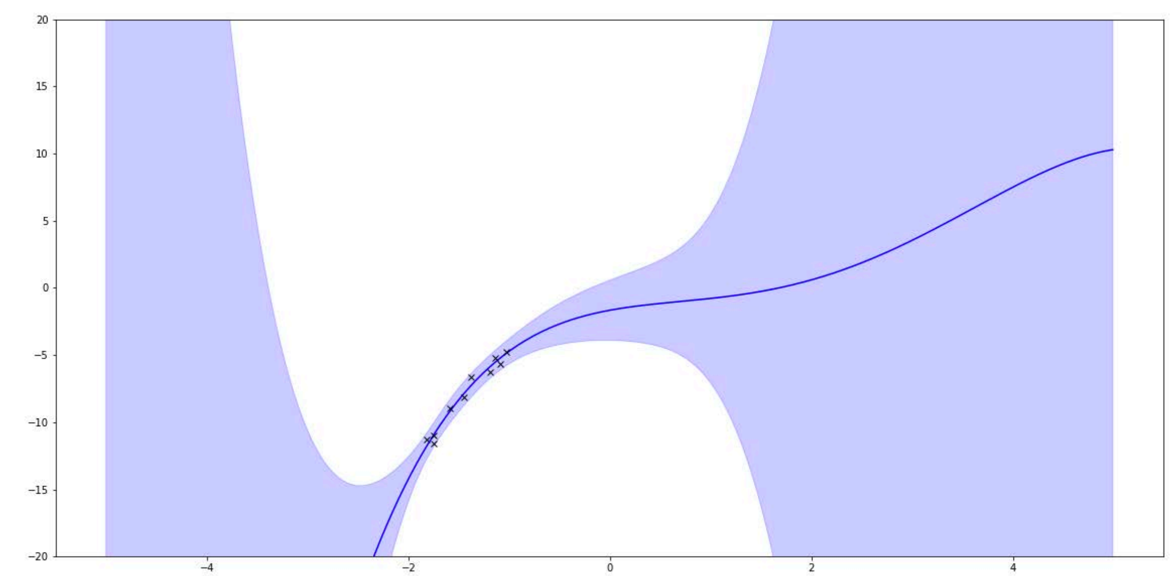
Figure 3.1 Examples of basis functions, showing polynomials on the left, Gaussians of the form (3.4) in the centre, and sigmoidal of the form (3.5) on the right.

Bayesian linear regression









Bayesian linear regression

```python
class BayesianLinearRegression:
  """
    Linear regression model: y = (w.T)*x + \epsilon
    w ~ N(0,beta^(-1)I)
    P(y|x,w) ~ N(y|(w.T)*x,alpha^(-1)I)
  """
  def __init__(self, X, y, alpha = 1.0, beta = 1.0):

      self.X = X
      self.y = y

      self.alpha = alpha
      self.beta = beta

      self.jitter = 1e-8


  def fit_MLE(self):
      xTx_inv = np.linalg.inv(np.matmul(self.X.T,self.X) + self.jitter)
      xTy = np.matmul(self.X.T, self.y)
      w_MLE = np.matmul(xTx_inv, xTy)

      self.w_MLE = w_MLE

      return w_MLE

  def fit_MAP(self):
      Lambda = np.matmul(self.X.T,self.X) + \
                (self.beta/self.alpha)*np.eye(self.X.shape[1])
      Lambda_inv = np.linalg.inv(Lambda)
      xTy = np.matmul(self.X.T, self.y)
      mu = np.matmul(Lambda_inv, xTy)

      self.w_MAP = mu
      self.Lambda_inv = Lambda_inv

      return mu, Lambda_inv

  def predictive_distribution(self, X_star):
      mean_star = np.matmul(X_star, self.w_MAP)
      var_star = 1.0/self.alpha + \
                 np.matmul(X_star, np.matmul(self.Lambda_inv, X_star.T))
      return mean_star, var_star
```