2a.

$$P(Y|X) = \frac{P(Y,X)}{P(X)}$$

$$P(Y = Xw_1 + w_0 + Z \mid X = x)$$

$$P(Z = Y - Xw_1 - w_0 \mid X = x) = P(Z = Y - Xw_1 - w_0)$$

$$= \frac{1}{\sqrt{2\pi}^1} \exp\left(\frac{-1}{2}(Y - Xw_1 - w_0)^2\right)$$

2b. MLE

$$Lik(w_1, w_0) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(Y - Xw_1 - w_0)^2\right)$$

$$L = \ln(Lik(w_1, w_0)) = \text{constants} - \frac{1}{2}\sum (Y - Xw_1 - w_0)^2$$

$$\arg \min_{w_1, w_0} \frac{1}{2}\sum (Y - Xw_1 - w_0)^2$$

$$\partial L/\partial w = \begin{cases} \partial L/\partial w_1 = \frac{1}{2} \cdot 2 \sum_{i=1}^{n} (Y_i - X_i w_1 - w_0) \cdot (-x_i) \quad (1) \\ \qquad = -\sum_{i=1}^{n} x_i (Y_i - Xw_1 - w_0) \\ \partial L/\partial w_0 = -\sum_{i=1}^{n} (Y_i - X_i w_1 - w_0) \quad (2) \end{cases}$$

Set equal to zero

(2) $\sum -Y_i + \sum X_i w + \sum w_0 = 0$

$$w_0 = \frac{1}{n}\sum (Y_i - X_i w)$$

① $-\sum_{i=1}^{n}(x_i y_i - x_i^2 w) + \sum x_i w_0 = 0$

$\sum x_i w_0 = \sum x_i y_i + x_i^2 w$, $w_0 = \dfrac{\sum(x_i y_i - x_i^2 w)}{\sum x_i}$

set equal sum for w_1

$$\frac{1}{n}\sum(y_i) - \frac{w}{n}\sum x_i = \frac{\sum(x_i y_i) - w\sum x_i^2}{n}$$

$$\frac{1}{n}\sum x_i^2\left[\sum(y_i) - w\sum x_i\right] = \sum(x_i y_i) - \frac{\sum x_i}{n} w \sum x_i^2$$

$$\frac{1}{n}\sum x_i^2 \sum y_i - \frac{w}{n}\sum x_i \sum x_i^2 = \sum(x_i y_i) - w\sum x_i^2$$

$$\frac{1}{n}\sum x_i^2 \sum y_i - \sum(x_i y_i) = \frac{w}{n}\sum x_i \sum x_i^2 - w\sum x_i^2$$

$$= w\left(\frac{\sum x_i \sum x_i^2}{n} - \sum x_i^2\right)$$

$$\left[w_1 = \frac{\frac{1}{n}\sum x_i^2 \sum y_i - \sum x_i y_i}{\frac{\sum x_i \sum x_i^2}{n} - \sum x_i^2} \right]$$

$$\left[w_0 = \frac{1}{n}\sum(y_i) - \frac{1}{n}\sum\left[x_i \cdot \frac{\frac{\sum x_i^2 \sum y_i}{n} - \sum x_i y_i}{\frac{\sum x_i \sum x_i^2}{n} - \sum x_i^2}\right]\right]$$

2.c. $z \sim U[-0.5, 0.5]$ — $\frac{1}{N}$ — $0.5 - (-0.5) = 1$

$P(Y|X=x) = P(z = Y - Xw \;;\; |X=x)$

$= P(z = Y - Xw) \quad P(Y_1 \dots Y_n | X_1 \dots X_n)$

$= \frac{1}{b-a} = \frac{1}{0.5+0.5} = 1$ $\quad z_i = y_i - x_i w$

$|z_i| \le 0.5$

$$F(z) = \begin{cases} 1 & -0.5 \le z_i \le 0.5 \\ 0 & \text{else} \end{cases}$$

I — Indicator: 0 if not in interval, 1 if in it

2.d.

$$\arg\max_w Lik(w) = \prod_{i=1}^{n} 1 \cdot I(z_1, \dots z_n \in [-0.5, 0.5])$$

$Lik = 1$ if $-0.5 \le \max(|z_1|, \dots, |z_n|) \le 0.5$

$Lik = 0$ if $\max(|z_1|, \dots, |z_n|) > 0.5$

$$\prod_{i=1}^{n} 1\{-0.5 \le y_i - x_i w \le 0.5\}$$

$$\arg\max_w \prod_{i=1}^{n} 1\left\{ \frac{0.5 + y_i}{x_i} \ge w \ge \frac{y_i - 0.5}{x_i} \right\} \quad \forall i = 1, \dots, n,\ x_i \neq 0$$

- y_i must be in case $x_i = 0$ $[-0.5 \le y_i \le 0.5]$

7.D cont.

· Need: one particular estimate for w

$$\frac{-0.5+y_i}{x_i} \leq w \leq \frac{0.5+y_i}{x_i}$$

Want tightest bound for w as our particular estimate

$$\text{largest } \underbrace{\frac{-0.5+y_i}{x_i}}_{(A)} \text{ from our Data}$$

& $\text{smallest } \underbrace{\frac{0.5+y_i}{x_i}}_{(B)}$ from our Data, such that $A < B$.

Our estimate should be with these tightest bound.

:) ;)

2.F. One > Ridge regression $\{(x_1,y_1),(x_2,y_2),\dots,(x_n,y_n)\}$

$y_i = x_i w + z_i \quad z_i \sim N(0,1)$ iid

$w \sim N(0,\sigma^2)$, x_i, $z \sim N(0,1)$ } independent

use Bayes

$Y_n | x_n, w \sim N(w^T x_n, \sigma^2$

$P(w \mid (x_1,y_1),\dots,(x_n,y_n))$

$[\vec{w} \sim N(0,\sigma^2) \Rightarrow \vec{w} = \sigma \vec{v} \quad \text{w/ } \vec{v} \sim N(0,1)]$

Bayes $P(w \mid D = \{(x_i,y_i)|_n\}) = \dfrac{P(\text{data}=D \mid w)\, P(w)}{P(\text{data}=D)}$

$\propto P(\text{data}=D \mid w)\, P(w)$

(proportional)

"posterior is proportional to prior · likelihood"

$P(w \mid x_1,y_1),(x_2,y_2),\dots,(x_n,y_n)) \propto P(w)\, P(x_1,\dots,x_n \mid w)$

$= \left(\prod_{i=1}^{n} P(y_i \mid x_i, w)\right) P(w \mid \sigma)$ ← Due to independence

$= P(w) \cdot \prod_{i=1}^{n} P(y_i \mid x_i, w)$

$P(y_i \mid x_i, w) = \dfrac{1}{(2\pi)^{1/2}} \exp\left\{-\frac{1}{2}(y_i - x_i w)^2\right\}$

$= P(z_i = y_i - x_i w \mid x_i w) =$

$P(w \mid \sigma) = P(w) = \dfrac{1}{(2\pi\sigma^2)^{1/2}} \cdot \exp\left(-\frac{1}{2\sigma^2}(w)^2\right)$

✡ as $e^{(y_1^2 - 2y_1x_1w + x_1^2w)} e^{(y_2^2 - 2y_2x_2w + x_2^2w)} \cdots = e^{\sum y_i^2} \cdot e^{w \sum 2x_iy_i} \cdot e^{w^2 \sum x_i^2}$

2.F (cont.)

$$\prod_{i=1}^{n} \left[\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(y_i - x_i w)^2\right) \cdot \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{w^2}{2\sigma^2}\right) \right]$$

$$= \frac{1}{(2\pi)^{n/2}} \frac{1}{(2\pi\sigma^2)^{n/2}} \prod \left[\exp\left(-\frac{1}{2}\left((y_i - x_i w)^2 - \frac{w^2}{\sigma^2}\right)\right) \right]$$

$\Updownarrow$ ✡

$$\exp\left[\frac{w^2}{2\sigma^2}\right] \exp\left[-\frac{1}{2}\left(\sum_{i=1}^{n} y_i^2 - 2w \sum x_i y_i + w^2 \sum x_i^2\right)\right]$$

$$\exp\left[-\frac{1}{2}\left(\sum y_i^2 - 2w \sum x_i y_i + w^2 \sum x_i^2 - \frac{w^2}{\sigma^2}\right)\right]$$

$$-\frac{1}{2\sigma^2}\left(\sigma^2 \sum y_i^2 - 2w\sigma^2 \sum x_i y_i + w^2\sigma^2 \sum x_i^2 - w^2\right)$$

$$-\frac{1}{2\sigma^2}\left(\underbrace{w^2\left(\sigma^2 \sum x_i^2 - 1\right)}_{a} + \underbrace{w\left(-2\sigma^2 \sum x_i y_i\right)}_{b} + \underbrace{\sigma^2 \sum y_i^2}_{c}\right)$$

$a(x+b)^2 + c$ $\qquad d = \frac{b}{2a} = \frac{-2\sigma^2 \sum x_i y_i}{2\sigma^2 \sum x_i^2 - 2}$

$$e = c - \frac{b^2}{4a} = \sigma^2 \sum y_i^2 - \frac{\left(2\sigma^2 \sum x_i y_i\right)^2}{4\sigma^2 \sum x_i^2 - 4}$$

$$-\frac{1}{2\sigma^2}\left[\left(\sigma^2 \sum x_i^2 - 1\right)\left(w + \frac{-2\sigma^2 \sum x_i y_i}{(2\sigma^2 \sum x_i^2 - 2)}\right)^2 + \underbrace{\sigma^2 \sum y_i^2 - \frac{\left(2\sigma^2 \sum x_i y_i\right)^2}{(4\sigma^2 \sum x_i^2 - 4)}}_{J}\right]$$

$$\frac{1}{(2\pi)^{1/2}} \cdot \frac{1}{(2\pi\sigma^2)^{1/2}} \cdot \exp\left(\frac{-1}{2\sigma^2} J\right) \exp\left[\frac{-1}{2\sigma^2}\left[(\sigma^2 \Sigma x_i^2 - 1]\left(w - \frac{2\sigma^2 \Sigma x_i y_i}{(2\sigma^2 \Sigma x_i^2 - 2)}\right)^2\right)\right]$$

$$\mu = \frac{\sigma^2 \hat{\Sigma} x_i y_i}{(\sigma^2 \hat{\Sigma} x_i^2) - 1)}$$

Gaussian

$$N(\mu, \sigma^2)$$

e.g. $y_i = w^T x_i + z_i$ $\quad y_i \in \mathbb{R}$, $w, x_i \in \mathbb{R}^d$

w Fixed

$z_i \sim N(0,1)$

MLE gives least squares

$$Lik(w) = p(y_i \mid w, x) = p(z_i = y_i - w^T x_i \mid w, x_i)$$

$$= p(z_1 \mid z_2 \dots z_n, w, x_i) \cdot p(z_2 \mid z_1, \dots, z_n, w, x_i)$$

$$= \hat{\prod_{i=1}} p(z_i \mid x_i, w)$$

$$= \hat{\prod_{i=1}} p(z_i = y_i - w^T x_i \mid x_i = x, w)$$

$$= \hat{\prod_{i=1}} \frac{1}{\sqrt{2\pi}^1} \exp\left(-\frac{1}{2}(y_i - w^T x_i)^2\right),$$

$$\arg\max_w \log(Lik(w)) = \text{constants} + \hat{\Sigma} -\frac{1}{2}(y_i - w^T x_i)^2$$

$$\Leftrightarrow \arg\min_w = \hat{\Sigma} \frac{(y_i - w^T x_i)^2}{2}$$

["Has same objective function as w/ OLS"
∴ it is clearly also the OLS estimator $\hat{w}$.

Currently: No Bonus

2.4 Multi Dimensional Ridge Regression

$\vec{w}$ random vector

$Y_i = w^T x_i + z_i \qquad Y_i \in \mathbb{R}, \vec{w}, x_i \in \mathbb{R}^d$

$z_i \sim N(0,1)$

$w_j \sim N(0, \sigma^2) \quad P(\vec{w}) = N(0, I\sigma^2)$

$$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} \quad Y = \begin{bmatrix} Y_1 \\ \vdots \\ Y_n \end{bmatrix} \Rightarrow X^T X = \sum x_i x^T \quad X^T Y = \sum x_i Y_i$$

Observe that $w^T x_i = x_i^T w$

$$\mathbb{P}(\vec{w}_j \mid (x_i, Y_i)_{i=1\dots d}) \propto \mathbb{P}((x_i, Y_i)_{i=1,\dots,d} \mid w_j)\mathbb{P}(w_j)$$

$$= \mathbb{P}(Y_i \mid x_i, w_j)\mathbb{P}(w_j) \quad \text{"by independence"}$$

$$= \mathbb{P}(z_i = Y_i - w^T x_i \mid x_i, w_j)\mathbb{P}(w_j)$$

$$\prod_{i=1}^{d} \mathbb{P}(w_i)\,\mathbb{P}(z_i = Y_i - w^T x_i)$$

$$= \prod_{i=1}^{d} \left[\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{1}{2}\frac{w_i^2}{\sigma^2} \right\} \cdot \frac{1}{\sqrt{2\pi}} \exp\left\{ -\frac{1}{2}(Y_i - w^T x_i)^2 \right\} \right]$$

$$= \frac{1}{\sqrt{2\pi\sigma^2}} \frac{1}{\sqrt{2\pi}} \prod_{i=1}^{d} \exp\left\{ -\frac{1}{2}\left(\frac{w_i}{\sigma^2} + (Y_i - w^T x_i)^2 \right) \right\}$$

$$\exp\left\{ -\frac{1}{2}\left(\frac{w_i}{\sigma^2} + Y_i^2 - 2w^T x_i Y_i + w^T x_i^2 \right) \right\} \exp\left\{ \left(\frac{w_2}{\sigma^2} + Y_2^2 - 2w^T x_2 Y_2 + w^T x_2^2 \right) \right.$$

$* \dots$

2.H cont.

$$[w_1\ w_2\ \dots\ w_n]^{1\times 2}$$

$$\exp\left\{-\frac{1}{2}\left(\sum y_i^2 - 2\sum w^T x_i y_i + \sum w^T x_i^2 + \frac{1}{\sigma^2}\sum w_i\right)\right\}$$

$$\exp\left\{-\frac{1}{2}\left(\underbrace{w^T{}^2\left(\sum x_i^2\right)}_{a} + w^T\underbrace{\left(-2\sum x_i y_i\right)}_{b} + \sum y_i^2 + \underbrace{\frac{1}{\sigma^2}\sum w_i}_{c}\right)\right\}$$

$d = \frac{b}{2a}$, $e = c - \frac{b^2}{4a}$

$a(x+d)^2 + e$

$$d = \frac{-2\sum x_i y_i}{2\sum x_i^2}$$

$$e = \left(\sum y_i^2 + \frac{1}{\sigma^2}\sum w_i\right) - \frac{4\left(\sum x_i y_i\right)^2}{4\sum x_i^2}$$

$$\exp\left\{-\frac{1}{2}\left(\sum x_i^2\left(w^T - \frac{\sum x_i y_i}{\sum x_i^2}\right)^2 + \sum y_i^2 + \frac{\sum w_i}{\sigma^2} - \frac{\left(\sum x_i y_i\right)^2}{\sum x_i^2}\right)\right\}$$

$$\mu = \frac{\sum x_i y_i}{\sum x_i^2} = \frac{X^T y}{\sum x_i^2}$$

3. a.

bias $E[\hat{x} - \mu]$ var $Var[\hat{x}]$

i) $\hat{x} = \frac{x_1 + x_2 + \dots + x_n}{n} = \mu$

$E[\hat{x} - \mu] = E[\hat{x}] - E[\mu] \Rightarrow E[\hat{x}] - \mu = 0$

$\frac{n\mu}{n} - \mu = 0$

ii) $\hat{x} = \frac{x_1 + \dots + x_n}{n+1}$

$E\left[\frac{x_1 + \dots + x_n}{n+1}\right] = \frac{1}{n+1}\left(E[x_1] + \dots + E[x_n]\right)$

$\frac{n \cdot \mu}{n+1}$ $\qquad \frac{n \cdot \mu}{n+1} - \mu = \mu\left(\frac{n}{n+1} - 1\right)$

iii) $\hat{x} = \frac{x_1 + \dots + x_n}{n + n_0}$ $\qquad \mu\left(\frac{n}{n+n_0} - 1\right)$

iv) $\hat{x} = 0 \qquad 0 - E[\mu] = -\mu$

3 b. i) $Var[\hat{x}] = Var\left(\frac{1}{n}(x_1 + \dots + x_n)\right) = \frac{n \cdot Var(x_i)}{n^2} = \frac{\sigma^2}{n}$

ii) $Var\left(\frac{1}{n+1}(x_1 + \dots + x_n)\right) = \frac{1}{(n+1)^2} n \, Var(x_i) = \frac{n\sigma^2}{(n+1)^2}$

iii) $Var\left(\frac{1}{n+n_0}(x_1 + \dots + x_n)\right) = \frac{1}{(n+n_0)^2} n\sigma^2 = \frac{n\sigma^2}{(n+n_0)^2}$

iv) $Var(\hat{x}) = E(\hat{x}^2) - E(\hat{x})^2 = 0$

x' is an iid copy of $\hat{x}$

3.c. $E[(\hat{x}-x')^2]$ $\quad \text{Var}(x) = E(x^2) - E(x)^2$

$E(\hat{x}^2 - 2\hat{x}x' + (x')^2)$

$E[\hat{x}^2] - 2E(\hat{x}x') + E[(x')^2]$

$\sigma_{\hat{x}}^2 + \hat{\mu}^2 - 2E(\hat{x})E(x') + \sigma_{x'}^2 + \mu^2$

$\sigma_{\hat{x}}^2 + \hat{\mu}^2 - 2\hat{\mu}\mu + \sigma_{x'}^2 + \mu^2$ $\quad = \hat{\sigma}^2 + \sigma^2 + (\hat{\mu}-\mu)^2$

$E[(\hat{x}-\mu)^2] = E[\hat{x}^2 - 2\hat{x}\mu + \mu^2]$

$E[\hat{x}^2] - 2E[\hat{x}]E[\mu] + E[\mu^2]$

$= \sigma_{\hat{x}}^2 + \hat{\mu}^2 - 2E[\hat{x}]E[\mu] + \mu^2$

$= \sigma_{\hat{x}}^2 + \hat{\mu}^2 - 2\hat{\mu}\mu + \mu^2$ $\quad = \hat{\sigma}^2 + (\hat{\mu}-\mu)^2$

- Our second derivation does not have variance. The x' introduces variance to the system.

$E(\hat{x}-\mu)=\mu$ $\quad E[(\hat{x}-\mu)^2] = \sigma_{\hat{x}}^2 + (\hat{\mu}-\mu)^2$ $\quad \frac{n}{n+1} - 1 = \frac{n-n-1}{n+1}$

3.d. i) $\frac{\sigma^2}{n} + 0^2$ $\quad \frac{n}{n+n_0} - 1 = \frac{n-n-n_0}{n+n_0} = \frac{-n_0}{n+n_0}$

ii) $\frac{n\sigma^2}{(n+n_0)^2} + \left(\frac{-n\mu}{n+n_0}\right)^2$ $\quad \frac{n^2}{(n+n_0)^2} \cdot n$

iii) $\frac{n\sigma^2}{(n+n_0)^2} + \left(\frac{-n\mu n_0}{n+n_0}\right)^2$

iv) $0 + (-\mu)^2 = \mu^2$

3.e $\hat{x} = \frac{x_1 + \ldots + x_n}{n+n_0}$ $\quad n_0$ samples of zero "dummies"

1) has $n_0 = 0$

2) has $n_0 = 2$

3) has $n_0 > n_0$

4) has $n_0 = \infty$

3.f. Bias as n_0 increase: our bias approaches $-\mu$
$F(x) \notin \{n_0\}$ increased (gets worse) approach constant
Var as n_0 increase: our variance approaches 0

3.9 $\lambda_0 = \alpha\lambda$ Solving for α to minimize

$$\text{Min } E[(\bar{x}-\mu)^2] = \frac{\lambda\sigma^2}{(\lambda+\lambda\alpha)^2} + \left(\frac{\mu\lambda\alpha}{\lambda+\alpha\lambda}\right)^2$$

$$\text{Min}\left[\frac{\lambda\sigma^2}{(\lambda+\lambda\alpha)^2} + \left(\frac{\mu\cdot\lambda\cdot\alpha}{\lambda+\alpha\lambda}\right)^2\right]$$

$$\mu^2\lambda^2\alpha^2(\lambda+\alpha\lambda)^{-2}$$

~~derivative~~

$$\frac{\partial}{\partial\alpha} = \lambda\sigma^2\left[-2(\lambda+\lambda\alpha)^{-3}\cdot\lambda\right] + \mu^2\lambda^2\left[2\alpha(\lambda+\lambda\alpha)^{-2} + \alpha^2(-2)(\lambda+\alpha\lambda)^{-3}\lambda\right]$$

derivative

$$2\left[\frac{\alpha\lambda\mu^2}{(\alpha+1)^3\lambda} - \sigma^2\right) = 0$$ solve for α

$$\frac{\alpha\lambda\mu^2}{(\alpha+1)^3\lambda} - \frac{\sigma^2}{(\alpha+1)^3\lambda} = 0$$

$$\frac{\alpha\lambda\mu^2}{(\alpha+1)^3\lambda} = \frac{\sigma^2}{(\alpha+1)^3\lambda}$$

$$\frac{\alpha\lambda\mu^2}{(\alpha+1)^3\lambda}\cdot(\alpha+1)^3 = \frac{\sigma^2}{\lambda}, \quad \boxed{\alpha = \frac{\sigma^2}{\lambda^2\mu^2}}$$

3.h $\alpha = \frac{\sigma^2}{n\mu^2}$ (μ small (α approaches ∞) real fast as μ is small & σ^2 appears a large #

3.I $\boxed{x' = x - \mu_0}$ s.t $E[x']$ small ~ 0

$E[x] - E[\mu_0] \sim 0$ implies it is close to μ_0

$\mu - \mu_0 \sim 0$

Bias ←

Variance ←→ $Var[x'] = Var[x] - Var[\mu_0]$

0

$= Var[x]$

3.J relation α & λ

- the regularization parameter reduces variance & also simultaneously increases bias.
- the greater our α the less impact our Data will have on our cls soln.
- λ serves to de the sand, regulary our Data
- Reg. most effective when we center our Data.
- Suggests that large values of α can reduce the variance as our prev. q showed

4. a.

Show $E[\hat{w}] = w^*$ $\quad \hat{w} = \arg\min_w \|y - Xw\|_2^2$

$\hat{w} = (X^T X)^{-1} X^T y$

$y = y^* + z_i$

$\hat{w} = (X^T X)^{-1} X^T (y^* + z)$

$E[(X^T X)^{-1} X^T y^*] + E[(X^T X)^{-1} X^T z]$

$E[(X^T X)^{-1} X^T y^*] + 0 = w^* = (X^T X)^{-1} X^T y^*$

Show that

(4) • $E[\|y^* - X\hat{w}\|_2^2] = y^{*T}y^* - y^{*T}Xw^* - w^{*T}X^T y^* + w^{*T}X^T X w^*$

$E[(y^* - X\hat{w})^T (y^* - X\hat{w})] =$

$E[(y^{*T} - \hat{w}^T X^T)(y^* - X\hat{w})] = E[y^{*T}y^* - y^{*T}X\hat{w} - \hat{w}^T X^T y^* + \hat{w}^T X^T X \hat{w}]$

(3) • $\|y^* - E[X\hat{w}]\|_2^2 = (y^* - E[X\hat{w}])^T (y^* - E[X\hat{w}])$

$(y^{*T} - w^{*T}X^T)(y^* - Xw^*) = [y^{*T}y^* - y^{*T}Xw^* - w^{*T}X^T y^* + w^{*T}X^T X w^*]$

4.a. (cont.)

(C) $= E\left[\| X\hat{w} - E(X\hat{w}) \|_2^2 \right]$

$= E\left[(X\hat{w} - E(X\hat{w}))^T (X\hat{w} - E(X\hat{w})) \right]$

$= E\left[(\hat{w}^T X^T - E(X\hat{w})^T)(X\hat{w} - E(X\hat{w})) \right]$

$= E\left[\hat{w}^T X^T X\hat{w} - \hat{w}^T X^T E(X\hat{w}) - E(X\hat{w})^T X\hat{w} + E(X\hat{w})^T E(X\hat{w}) \right]$

$= E\left[\hat{w}^T X^T X\hat{w} \right] -$

$w^{*T} X^T X w^* - w^{*T} X^T X w^* - w^{*T} X^T X w^* + w^{*T} X^T X w^*$

$= 0$

$A = B + C \qquad C = 0$

$\therefore$

$A = B$ which is what we set out to show on the previous page.

4.6. $v \sim N(0, \Sigma)$ $\quad \Sigma \in \mathbb{R}^{d \times d}$, $A \in \mathbb{R}^{m \times d}$

$Av \sim N(0, A\Sigma A^T)$

Show $\quad \hat{w} \sim N(w^*, \sigma^2 (X^T X)^{-1})$

$z \sim N(0, \sigma^2)$ $\qquad \hat{w} = (X^T X)^{-1} X^T (y^* + z)$

$\hat{w} = (X^T X)^{-1} X^T y^* + (X^T X)^{-1} X^T z$

$\hat{w} = w^* + (X^T X)^{-1} X^T z$

$X^T z \sim N(0, X^T \sigma^2 X)$

$w^* + X^T z \sim N(w^*, X^T \sigma^2 X)$

$w^* + (X^T X)^{-1} X^T z \sim N(w^*, (X^T X)^{-1} X^T \sigma^2 X ((X^T X)^{-1})^T) + w^*$

$\therefore (X^T X)^{-1} X^T \sigma^2 X ((X^T X)^{-1})^T = \sigma^2 (X^T X)^{-1}$

$X^{-1} X^{T-1} X^T \sigma^2 X (X^{-1} X^{T-1})^T$

$\underbrace{X^{-1} X^{T-1} X^T}_{I} \sigma^2 \underbrace{X (X^{T-1} X^{-1})^T}_{I} = \sigma^2 (X^T X)^{-1}$

$\sigma^2 (X^{-1} X^{T-1})$

$X^{-1} \sigma^2 X^{-1T} = \sigma^2 X^{-1} X^{T-1}$ —

σ^2 is a scalar thus the above expression is true equivalent. $\blacksquare$

$N \sim (w^*, \sigma^2 (X^T X)^{-1})$

$\text{Var}(\hat{w}|\beta) = (\sigma^2)(X^TX)^{-1}$ $\quad \frac{1}{n}E[\|X\hat{w} - Xw^*\|_2^2] = \frac{\sigma^2 d}{n}$

4.c. IF $\beta \sim N(\mu, \Sigma)$

Math Stack exchange

$\Rightarrow E(\|\beta\|_2^2) = \|\mu\|_2^2 + tr(\Sigma)$

∴ Let us first find $0 = E(X\hat{w} - Xw^*)$,

$= XE(\hat{w}) - Xw^* = Xw^* - Xw^* = 0$

So $E(\|X\hat{w} - Xw^*\|_2^2) = \|0\|_2^2 + tr(\Sigma) = tr(\Sigma)$

where Σ is variance of $\underbrace{X\hat{w} - Xw^*}$

"Xw^* is a constant"

$\text{Var}(X\hat{w} - Xw^*) = \text{Var}(X\hat{w})$ $\quad X^2\sigma^2(X^TX)^{-1}$

$\sigma^2 X^2(X^T$

$\text{Var}(X\hat{w}) = X^TX\,\text{Var}(\hat{w}) = X^TX\sigma^2(X^TX)^{-1}$

$tr(X^TX\sigma^2(X^TX)^{-1}) = \sigma^2 tr(X^TX(X^TX)^{-1})$

$= \sigma^2 \cdot tr(I) = \sigma^2 \cdot d$

$\frac{1}{n}E(\|X\hat{w} - Xw^*\|_2^2) = \frac{1}{n}\left(\|0\|_2^2 + tr(\text{Var}(X\hat{w} - Xw^*))\right)$

$= \boxed{\frac{\sigma^2 d}{n}}$

$\{x_i\}_{i=1}^n$ (1) $y = y^* + z$

u.d. $\{a_i, y_i\}_{i=1}^n$ $y_i = w_1 x_i + w_0 + z_i$

χ w/ $D+2$ polynomial Features $P_D(x_i) = [1, x_1, \ldots, x_i^D]$

≥ 2

$w^* = (\chi^T \chi)^{-1} \chi^T y^*$

$y_i = \underbrace{w_1 x_i + w_0}_{y^*} + z_i$ $\quad y_i = y^* + z_i$

$y_i^* = w_1 x_i + w_0$

$\hat{w} = (\chi^T \chi)^{-1} \chi^T y = (\chi^T \chi)^{-1} \chi^T (w_1 x_i + w_0)$

$= (\chi^T \chi)^{-1} \chi^T$

bias

$\|y^* - \chi w^*\|_2^2 = 0$

b/c $\chi w^* = y^*$

$w^* = \begin{bmatrix} w_0 \\ w_1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$ a $d+1 \times 1$ vector

avg expected prediction squared error

$\frac{1}{n} \mathbb{E}\left[\|\chi\hat{w} - \chi w^*\|_2^2\right]$ Usg M.C $= \frac{\sigma^2(D+1)}{n} \leq \varepsilon$

$n \geq \frac{\sigma^2(D+1)}{\varepsilon}$

HW03 - CS 189

1.
Who else did you work with on this homework? In case of course events, just describe the group. How did you work on this homework? Any comments about the homework?

I worked on this homework with Ehimare Okoyomon, Prashanth Ganeth, and Daniel Mockaitis. We worked by getting together throughout the week and communicating on facebook.

I certify that all solutions are entirely in my words and that I have not looked at another student's solutions. I have credited all external sources in this write up}

Nicholas Lorio, 26089160

Question 6.

What time is it?

9:57PM

Sources

https://ocw.mit.edu/courses/mathematics/18-443-statistics-for-applications-fall-2006/lecture-notes/lecture2.pdf

In [167]:
```
import numpy as np
import matplotlib.pyplot as plt
import pickle
```

HW03

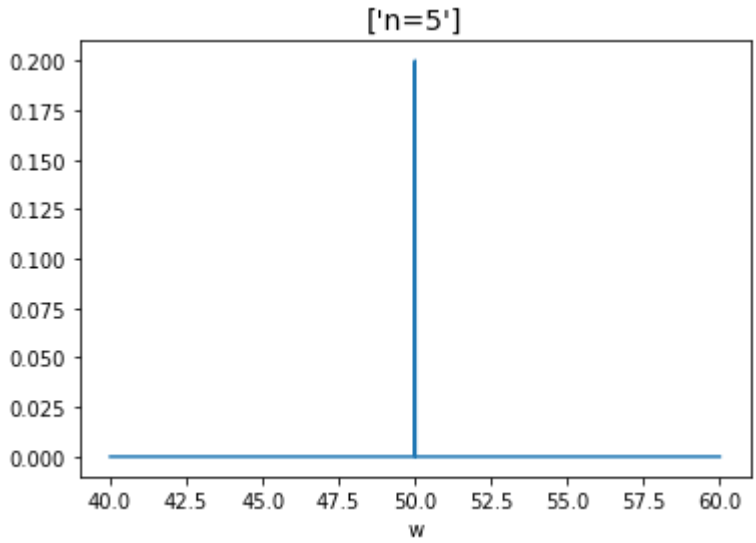
Question 2

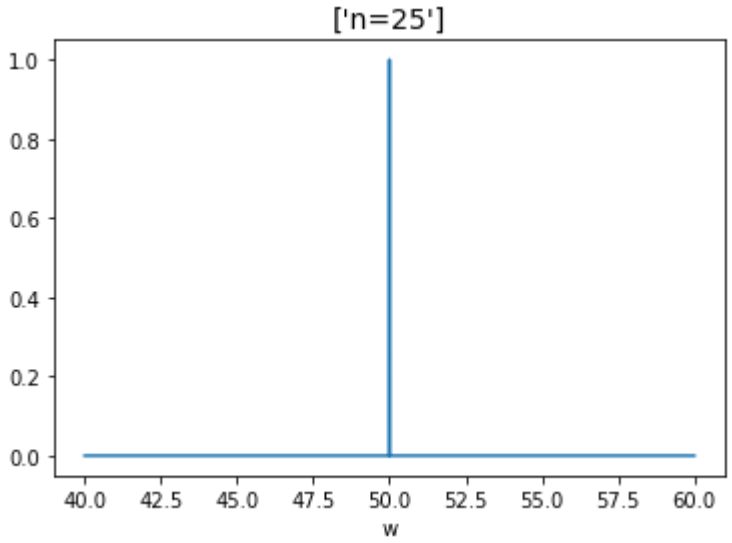
2.E

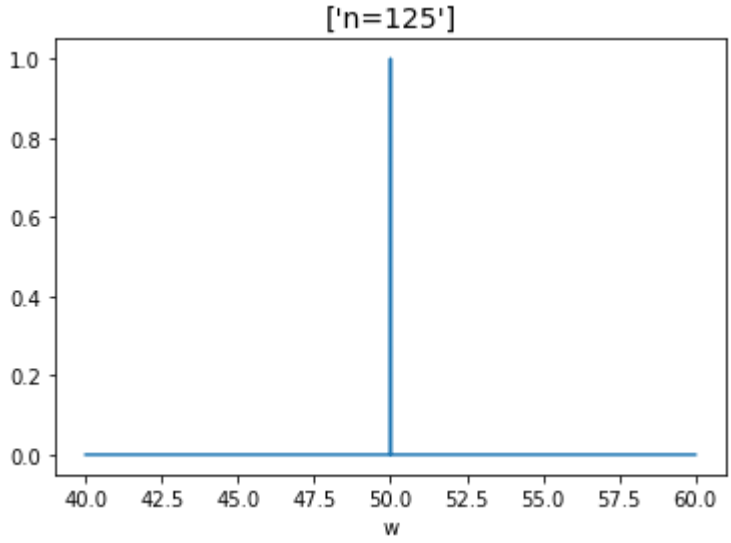
In [55]:
```
import numpy as np
import matplotlib.pyplot as plt

sample_size = [5,25,125,625]
plt.figure(figsize=[12, 10])
low_bound = -0.5
high_bound = 0.5
N = 10001
W = np.linspace(40, 60, num=N)
w_true = 50
print(W)

for k in range(4):
    n = sample_size[k]

    # generate data
    # np.linspace, np.random.normal and np.random.uniform might be usef
ul functions
    Xs = np.linspace(1, 100, num=n)
    Ys = np.array([x*w_true + np.random.uniform(low_bound, high_bound)
for x in Xs])


    likelihood = np.ones(N) # likelihood as a function of w

    for i in range(N):
        w_i = W[i]
        in_bound = True
        for j in range(n):
            y_j = Ys[j]
            x_j = Xs[j]

            if w_i > (y_j + 0.5)/x_j or w_i < (y_j - 0.5)/x_j:
                in_bound = False
                break
        if in_bound:
            likelihood[i] = 1
            print("Likelihood is 1 for w equal to " + str(w_i) + " in s
ample" + str(n))
        else:
            likelihood[i] = 0

        # compute likelihood
    #print(sum(likelihood))
    likelihood /= sum(likelihood) # normalize the likelihood

    plt.figure()
    # plotting likelihood for different n
    plt.plot(W, likelihood)
    plt.xlabel('w', fontsize=10)
    plt.title(['n=' + str(n)], fontsize=14)

plt.show()
```

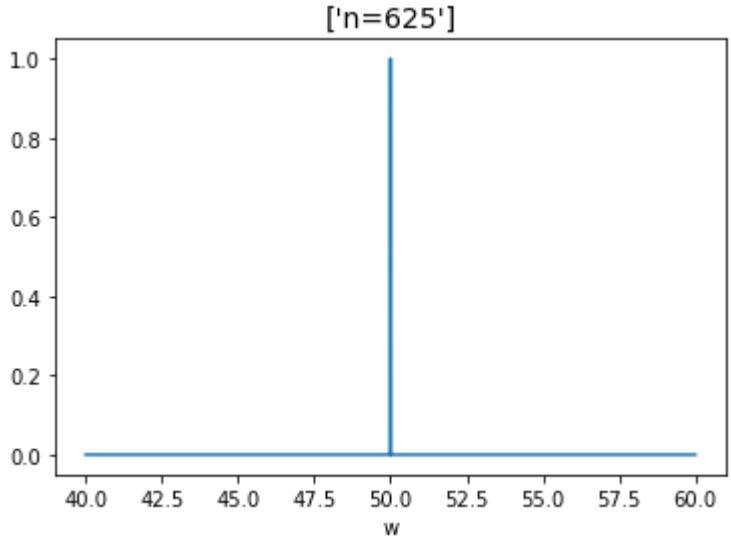
```
[40.    40.002 40.004 ... 59.996 59.998 60.   ]
Likelihood is 1 for w equal to 49.994 in sample5
Likelihood is 1 for w equal to 49.996 in sample5
Likelihood is 1 for w equal to 49.998 in sample5
Likelihood is 1 for w equal to 50.0 in sample5
Likelihood is 1 for w equal to 50.002 in sample5
Likelihood is 1 for w equal to 50.0 in sample25
Likelihood is 1 for w equal to 50.0 in sample125
Likelihood is 1 for w equal to 50.0 in sample625
```

```
<matplotlib.figure.Figure at 0x7f7c3fea72e8>
```









As n gets large, the amount of w estimates that accurately fit witin our bounds decreases. The MLE of the uniform distribution can be either 1 or zero. As n increases our estimation for the model becomes less general and the variance of our MLE parameter decreases.

2.I

In [174]:
```
import numpy as np
import matplotlib.pyplot as plt

sample_size = [5,25,125]
w0_true = 20
w1_true = 50
N = 10001
w0 = np.linspace(10,30, num=N)
w1 = np.linspace(40, 60, num=N)
Xs = np.random.rand(n,2)
Ys = np.array([x[0]*w0_true + x[1]*w1_true + np.random.normal(0, 1) for
 x in Xs])

for k in range(4):
    n = sample_size[k]

    # generate data
    # np.linspace, np.random.normal and np.random.uniform might be usef
ul functions




    # compute likelihood

    N = 1001
    # W0s =
    # W1s =
    likelihood = np.ones([N,N]) # likelihood as a function of w_1 and w
_0

    for i1 in range(N):
        # w_1 = W1s[i1]
        for i2 in range(N):
            # w_2 = W2s[i2]
            for i in range(n):
                # compute the likelihood here

    # plotting the likelihood
    plt.figure()
    # for 2D likelihood using imshow
    plt.imshow(likelihood, cmap='hot', aspect='auto',extent=[0,4,0,4])
    plt.xlabel('w0')
    plt.ylabel('w1')
    plt.show()
    print(n)
```

```
  File "<ipython-input-174-edcd92a9c715>", line 38
    plt.figure()
    ^
IndentationError: expected an indented block
```

Question 4

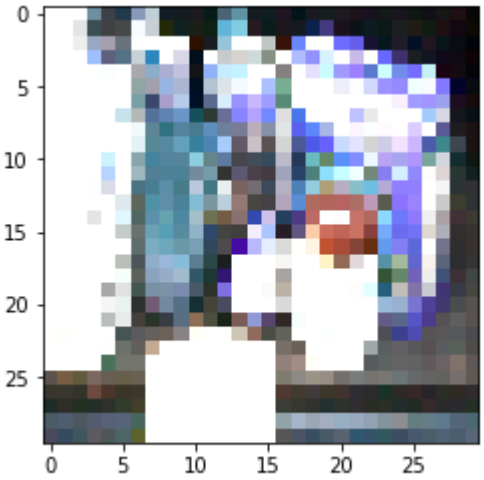
4.E

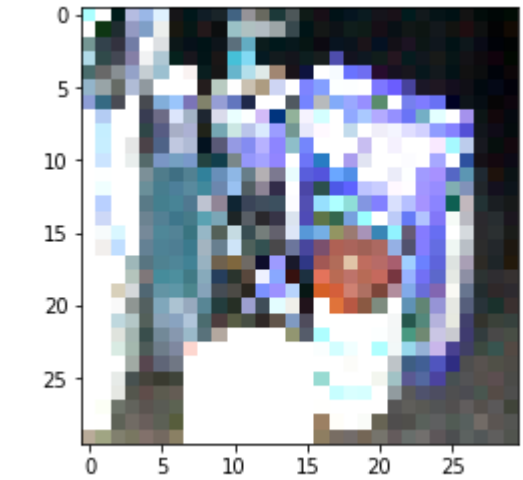
In [173]:
```
# assign problem parameters
w0=1
w1=1
N = [10, 100, 1000]
# generate data
# np.random might be useful
alpha = []
z = []
D = 10
for i in range(D):
    alpha.append(np.random.uniform(-1, 1))
    z.append(np.random.normal(0, 1))

y = np.array(alpha)*w1 + w0 + z

err = np.zeros(D)
for d in range(D):
    Xs = np.array([[x**i for i in range(d)] for x in alpha])
    w = np.linalg.lstsq(Xs, y)[0]
    err[d] = np.sum(np.square(Xs.dot(w) - y)) / len(y)


plt.figure()
# plotting likelihood for different n
plt.plot(D, err)
plt.xlabel('D', fontsize=10)
plt.ylabel('Err', fontsize=10)
plt.title(['n=' + str(n)], fontsize=14)



# fit data with different models
# np.polyfit and np.polyval might be useful


# plotting figures
# sample code

# plt.figure()
# plt.subplot(121)
# plt.semilogy(np.arange(1, deg+1),error[:,-1])
# plt.xlabel('degree of polynomial')
# plt.ylabel('log of error')
# plt.subplot(122)
# plt.semilogy(np.arange(n_s, n_s+step), error[-1,:])
# plt.xlabel('number of samples')
# plt.ylabel('log of error')
# plt.show()
```

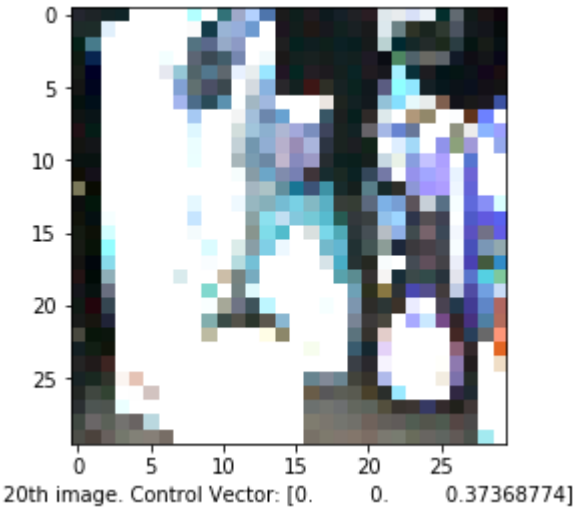
```
---------------------------------------------------------------------------
-----
LinAlgError                               Traceback (most recent call
 last)
<ipython-input-173-6e9650117cd2> in <module>()
     17 for d in range(D):
     18     Xs = np.array([[x**i for i in range(d)] for x in alpha])
---> 19     w = np.linalg.lstsq(Xs, y)[0]
     20     err[d] = np.sum(np.square(Xs.dot(w) - y)) / len(y)
     21

/usr/lib/python3.6/site-packages/numpy/linalg/linalg.py in lstsq(a, b,
 rcond)
   1976         b = b[:, newaxis]
   1977     _assertRank2(a, b)
-> 1978     _assertNoEmpty2d(a, b)  # TODO: relax this constraint
   1979     m  = a.shape[0]
   1980     n  = a.shape[1]

/usr/lib/python3.6/site-packages/numpy/linalg/linalg.py in _assertNoEm
pty2d(*arrays)
    223     for a in arrays:
    224         if _isEmpty2d(a):
--> 225             raise LinAlgError("Arrays cannot be empty")
    226
    227 def transpose(a):

LinAlgError: Arrays cannot be empty
```

Question 5

In [75]:
```
class HW3_Sol(object):


    def __init__(self):
        pass

    def load_data(self):
        self.x_train = pickle.load(open('x_train.p','rb'), encoding='la
tin1')
        self.y_train = pickle.load(open('y_train.p','rb'), encoding='la
tin1')
        self.x_test = pickle.load(open('x_test.p','rb'), encoding='lati
n1')
        self.y_test = pickle.load(open('y_test.p','rb'), encoding='lati
n1')


hw3_sol = HW3_Sol()
hw3_sol.load_data()

#please visualize the 0th, 10th and 20th images in the training datase
t.
# Also find out what's their corresponding control vectors.

#Data
    # Tuples of n values
    # First index is sample of n
    # Second index is row of sample
    # Third index is column of a row
    # Last index is an entry in RGB pixel
    #Shape of x_train (91, 30, 30, 3)



#5.a
def visualize_control_vectors(index):
    sample = hw3_sol.x_train[index]
    y_sample = hw3_sol.y_train[index]
    plt.imshow(sample)
    plt.xlabel(str(index) + "th image. Control Vector: " + str(y_sample
))
    plt.show()

visualize_control_vectors(0)
visualize_control_vectors(10)
visualize_control_vectors(20)
```


0
5
10
15
20
25
0th image. Control Vector: [0. -1. 0.]


0
5
10
15
20
25
10th image. Control Vector: [-1. -0.45111084 -1.]


0
5
10
15
20
25
20th image. Control Vector: [0. 0. 0.37368774]

In [156]:
```
#5.b
X = np.vstack([sample.flatten() for sample in hw3_sol.x_train])
U = hw3_sol.y_train

w = np.linalg.lstsq(X, U, rcond=None)
print("PI = ")
print(w[0])
```

```
PI =
[[2.89426558e-05 6.73236427e-05 9.28172481e-05]
 [4.33752719e-05 6.97139943e-05 6.69960177e-05]
 [4.24503660e-05 5.92797801e-05 6.84843436e-05]
 ...
 [1.36854182e-05 8.81609378e-05 5.15078903e-05]
 [1.96098789e-06 8.02702569e-05 3.78611891e-05]
 [7.46474147e-07 7.25823830e-05 3.73556513e-05]]
```

5.b

The weights are on the order of e^(-5) to e^(-7). However, some of the weights are very large and some of the weights are very small in relation to eachother. There is a large difference in weight values due to the relatively large differences in the pixel values of our data. We can amend this through the implementation of standardization and ridge regression.

In [98]:
```
#5.c
def lstsqL(A, b, lambda_):
     return np.linalg.solve((A.T @ A) + np.eye(len(A[0]))*lambda_, A.T
@ b)
lambda_array = [0.1, 1.0, 10, 100, 1000]

for l in lambda_array:
    w_hat = lstsqL(X, U, l)
    err = np.average(np.square(X.dot(w_hat) - U))
    print(str(err) + "      Traning Error for Lambda: " + str(l))

print()
print("This is different than Ehimares, Confirm")
```

```
29320754873.53885      Traning Error for Lambda: 0.1
134705132893.04636      Traning Error for Lambda: 1.0
2439924413.431048      Traning Error for Lambda: 10
3207855552.0886855      Traning Error for Lambda: 100
7188177320.168516      Traning Error for Lambda: 1000
```

In [154]:
```
#5.d


X_S = (X/255)*2 - 1 # BUT THESE DATA POINTS ARE NOT ALL BETWEEN 0 and 1
 ?!
print("Errors for standardized X")
print()
for l in lambda_array:
    w_hat = lstsqL(X_S, U, l)
    err = np.average(np.square(X_S.dot(w_hat) - U))
    print(str(err) + "      Traning Error for Lambda: " + str(l))
```

```
Errors for standardized X

1.1413557810626031e-07      Traning Error for Lambda: 0.1
1.0168794432272311e-05      Traning Error for Lambda: 1.0
0.0005420243232873813      Traning Error for Lambda: 10
0.011143580331285194      Traning Error for Lambda: 100
0.0814696917144622      Traning Error for Lambda: 1000
```

In [155]:
```
#5.e
Xtest = np.vstack([sample.flatten() for sample in hw3_sol.x_test])
Utest = hw3_sol.y_test

for l in lambda_array:
    w_hat = lstsqL(Xtest, Utest, l)
    err = np.average(np.square(Xtest.dot(w_hat) - Utest))
    print(str(err) + "      Test Error for Lambda: " + str(l))
print()
print()
X_S_test = (Xtest/255)*2 - 1
for l in lambda_array:
    w_hat = lstsqL(X_S_test, Utest, l)
    err = np.average(np.square(X_S_test.dot(w_hat) - Utest))
    print(str(err) + "      Standardized Test Error for Lambda: " + str
(l))
```

```
1808276541.284668      Test Error for Lambda: 0.1
30271423502.21572      Test Error for Lambda: 1.0
2270580651.7596126      Test Error for Lambda: 10
2041647982.9062707      Test Error for Lambda: 100
954418744.3903918      Test Error for Lambda: 1000


1.82861668664678e-08      Standardized Test Error for Lambda: 0.1
1.7546293832295183e-06      Standardized Test Error for Lambda: 1.0
0.00012689114816606716      Standardized Test Error for Lambda: 10
0.0037786682640806486      Standardized Test Error for Lambda: 100
0.04386417414236367      Standardized Test Error for Lambda: 1000
```

5.e.

Minimum error in test data OLS Ridge Regression with lambda = 100 for both standardized and unstandardized test data.

Lambda affect on performance in terms of bias:

As lambda increases our bias increases.

Lambda affect on performance in terms of variance:

As lambda increases our variance decreases.

In [157]:
```
#5.f

#With training data


without = \
np.linalg.svd((X.T @ X) + 100*np.eye(len(X[0])), compute_uv = False)
k = np.max(without) / np.min(without)
print(k)

with_ = \
np.linalg.svd((X_S.T @ X_S) + 100*np.eye(len(X_S[0])), compute_uv = Fal
se)
k = np.max(with_) / np.min(with_)
print(k)
```

```
1626465.0894416121
440.7329578145653
```

Question 6.

Q: Regarding question 5, robotic learning of control from demonstrations and images, how would higher resolution visuals in our data affect our prediction error and the performance of Huey?

A: We have already seen and proven how more data can improve our model. Increasing the resolution of our visuals would increase the amount of good features that we have in our model. Sharper images would provide greater distinction between the data points, thus we would have more singular values that are discrete. In particular, we can note that better features would be those that better define the outlines of the objects. As such sharper images could potentially overcomplicate\ our model as our goal is to recognize the boundaries between object and not the specific details of each object. Features which give greater detail to the inner boundary portions of the objects would not serve as good features.

We can interpret this as something to keep in mind when designing machine learning models, our task can determine what data we need to collect for training.