THE UNIVERSITY OF SYDNEY

Advanced Machine Learning (COMP 5328)

Learning with Noisy Data: On the Robustness of Surrogate Loss Functions

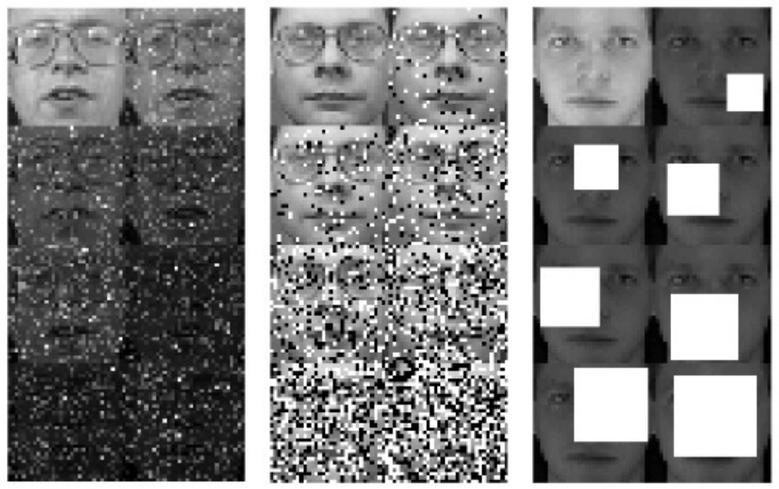
Tongliang Liu

THE UNIVERSITY OF SYDNEY

Noise on the observations

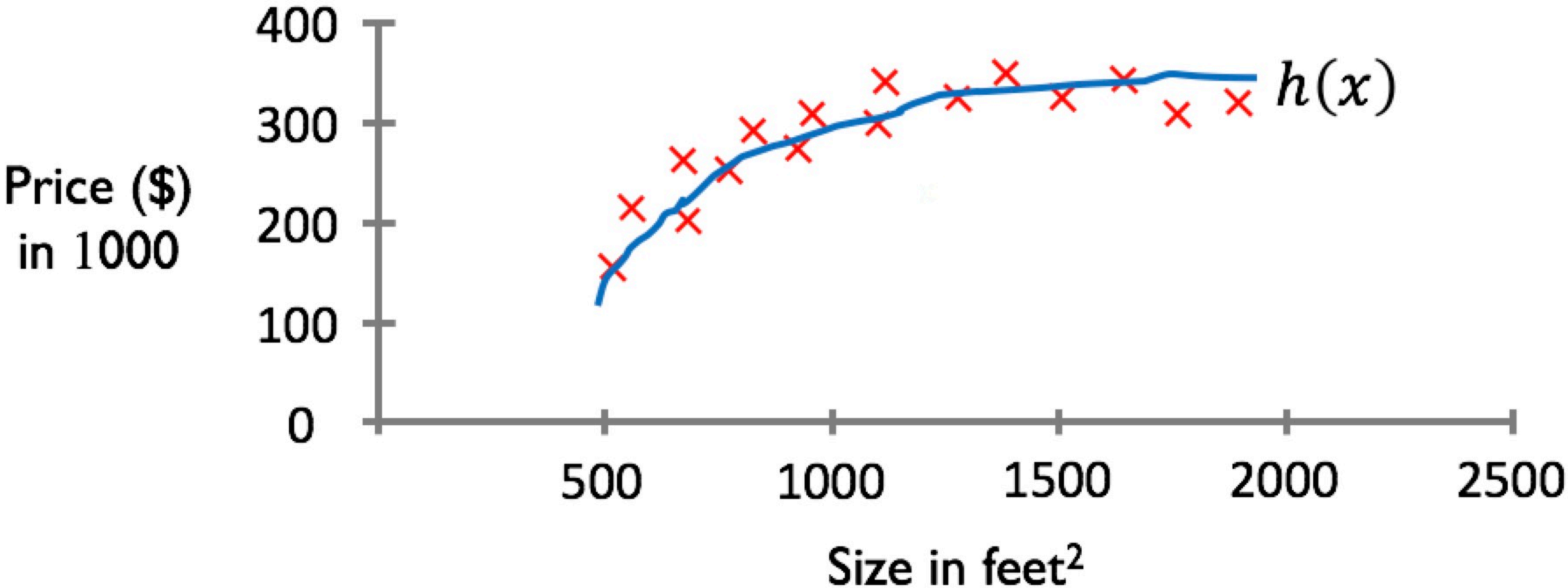
(a) (b) (c)

Ma, Kede, et al. "dipIQ: Blind image quality assessment by learning-to-rank discriminable image pairs." IEEE Transactions on Image Processing 26.8 (2017): 3951-3964.

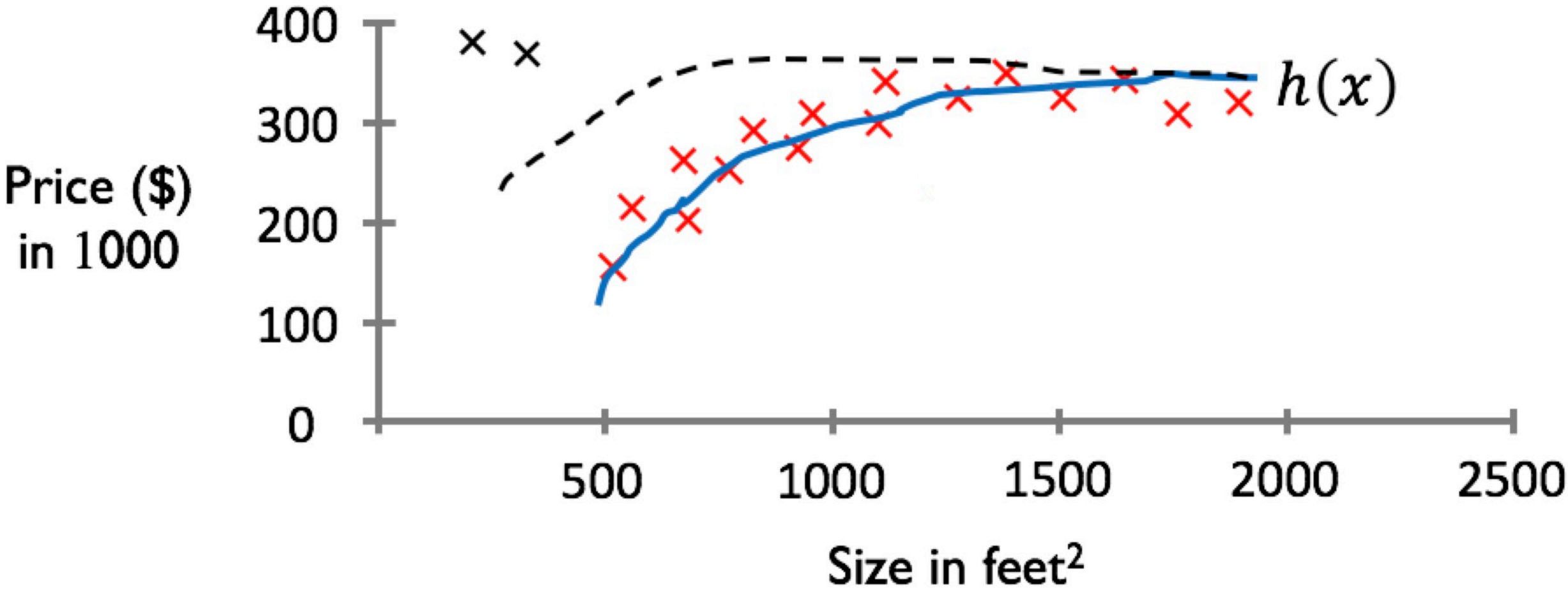


Guan, Naiyang, et al. "Truncated Cauchy Non-negative Matrix Factorization for Robust Subspace Learning." IEEE Transactions on Pattern Analysis and Machine Intelligence (2017).

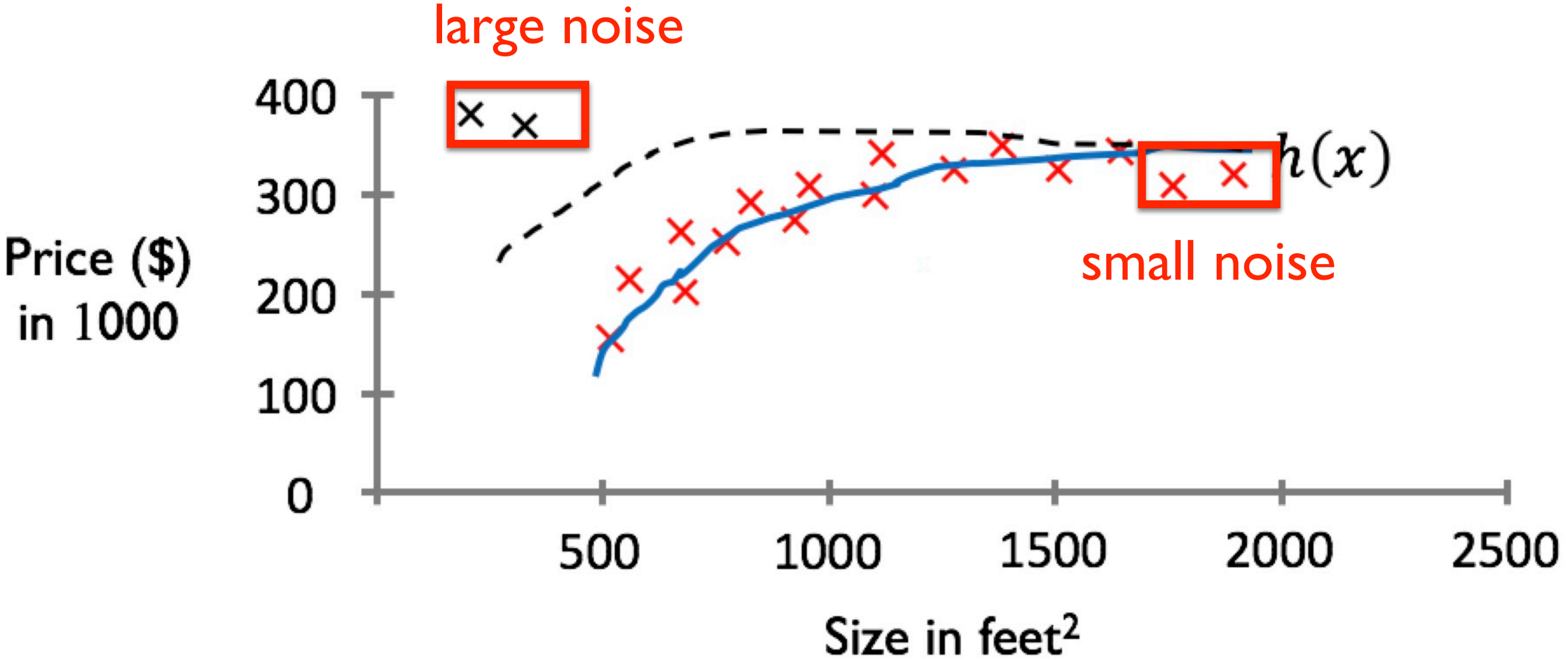
Small noise vs large noise



Small noise vs large noise



Small noise vs large noise



Why we need to exploit noisy data?

THE UNIVERSITY OF SYDNEY

- The explosion of data obtained through, e.g., social networks or special data-collecting channels within organisations and sensor networks
- Big data can easily be corrupted by noise during collection, convection and combination due to its highly distributed, dynamic and unstructured nature.

How to deal with noisy data?

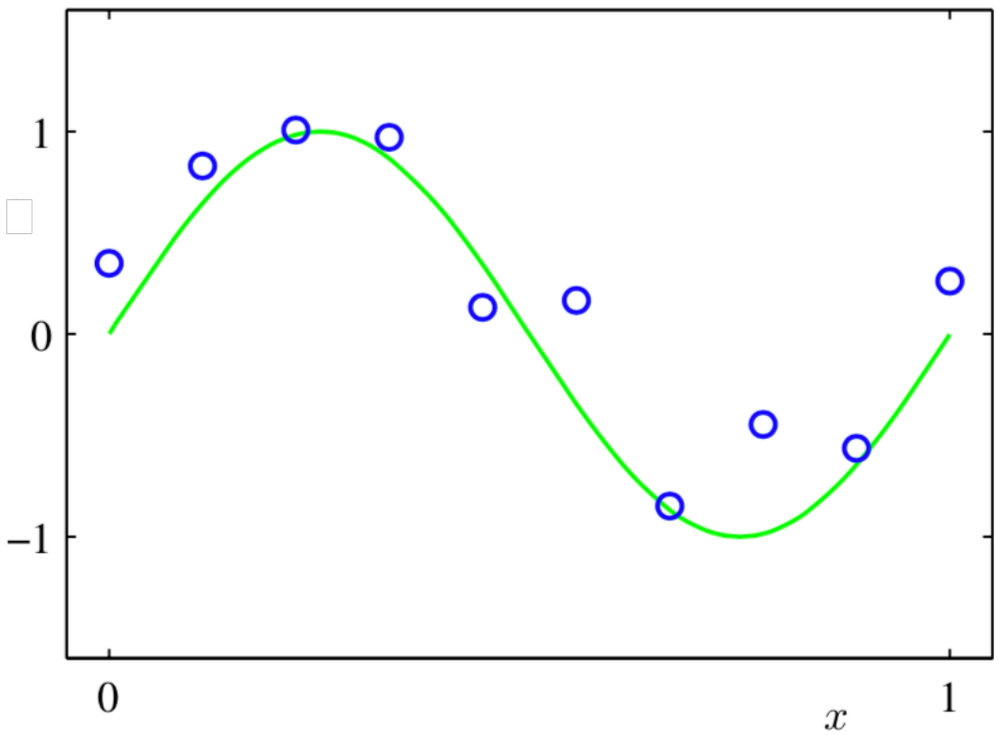
THE UNIVERSITY OF SYDNEY

- Cleanse the data
- Design machine learning models to be robust to the noise

THE UNIVERSITY OF SYDNEY

Review of linear regression: A probabilistic perspective

Linear regression with Gaussian noise



- $y = h(x) + \epsilon.$
- $\epsilon \sim \mathcal{N}(0, \beta^{-1})$
- $y|x \sim \mathcal{N}(h(x), \beta^{-1})$

Bishop's book: "Pattern Recognition and Machine Learning"

Bayes' rule

$$p(\theta|S) = \frac{p(S|\theta)p(\theta)}{p(S)}$$

Likelihood, prior, and posterior

THE UNIVERSITY OF SYDNEY

- **Likelihood** of θ is the probability of observing the data given θ. Given the data sample S, we denote the likelihood as $p(S|\theta)$.
- **Prior** of θ is a distribution that describes any prior beliefs of θ. We denote the prior as $p(\theta)$.
- **Posterior** of θ is proportional to the likelihood times the prior. We denote the posterior as $p(\theta|S)$.

Maximum Likelihood Estimation (MLE)

According to the i.i.d. assumption, the likelihood function is rewritten as

$$p(S|\theta) = \prod_{i=1}^{n} p(x_i, y_i|\theta).$$

Sometimes, we also define the likelihood as follows

$$p(S|\theta) = \prod_{i=1}^{n} p(y_i|x_i, \theta).$$

Maximum Likelihood: Find the value of θ maximising the likelihood $p(S|\theta)$, i.e., it is the value of θ that makes the observed data the "most probable".

Maximum A Posterior (MAP)

THE UNIVERSITY OF SYDNEY

Bayes' rule

$$P(\theta|S) = \frac{P(S|\theta)P(\theta)}{P(S)}$$

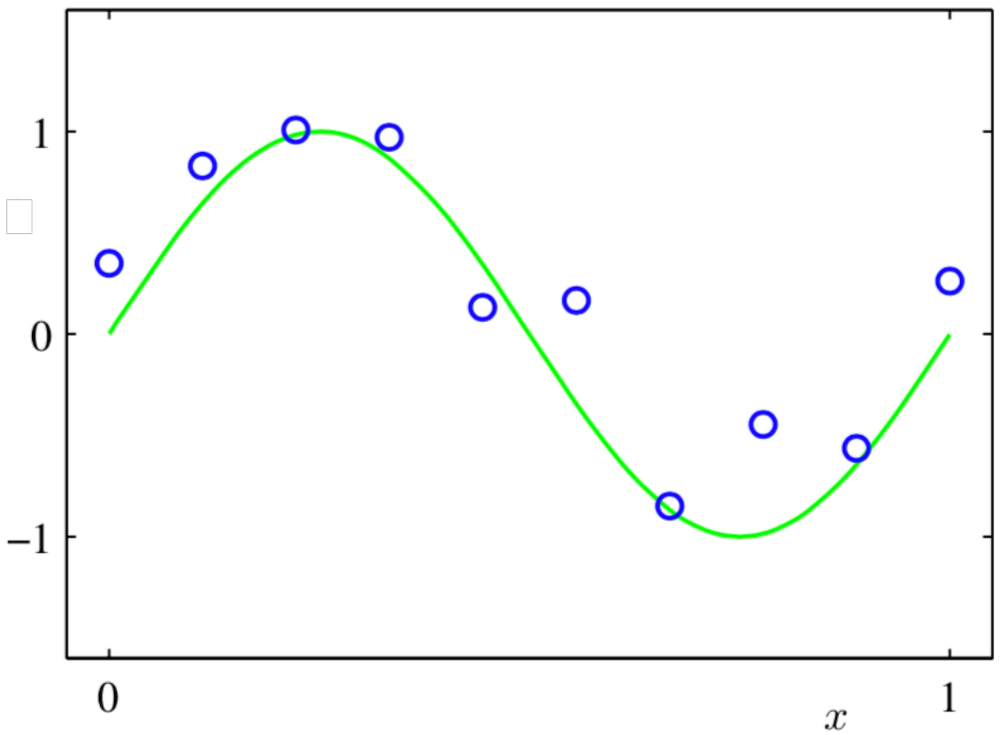
$$P(\theta|S) \propto P(S|\theta)P(\theta)$$

$$\arg\max_{\theta} P(\theta|S) = \arg\max_{\theta} P(S|\theta)P(\theta)$$

$$\arg\min_{\theta}(-\log P(\theta|S)) = \arg\min_{\theta}(-\log P(S|\theta) - \log P(\theta))$$

Maximum Posterior $P(\theta|S)$: the observed data makes the value of θ the "most probable".

Linear regression with Gaussian noise



- $y = h(x) + \epsilon.$
- $\epsilon \sim \mathcal{N}(0, \beta^{-1})$
- $y|x \sim \mathcal{N}(h(x), \beta^{-1})$

Bishop's book: "Pattern Recognition and Machine Learning"

Modelling Noisy Observations

Lets assume data is from a deterministic function with additive Gaussian noise.

$$y = h(x) + \epsilon \qquad \text{with} \qquad \epsilon \sim \mathcal{N}(0, \beta^{-1})$$

Equivalently,

$$p\{y|x, h, \beta\} = \mathcal{N}(y|h(x), \beta^{-1})$$

Given the training data: $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$

The expression of the likelihood the model is:

$$p(S|X, h, \beta^{-1}) = \prod_{i=1}^{n} \mathcal{N}(y_i|h(x_i), \beta^{-1})$$

Maximum Likelihood

$$p(S|X,h,\beta^{-1}) = \prod_{i=1}^{n} \mathcal{N}(y_i|h(x_i),\beta^{-1})$$

$$= \prod_{i=1}^{n} \sqrt{\frac{\beta}{2\pi}} \exp\left(-\frac{\beta(y_i - h(x_i))^2}{2}\right)$$

$$= \left(\frac{\beta}{2\pi}\right)^{n/2} \prod_{i=1}^{n} \exp\left(-\frac{\beta(y_i - h(x_i))^2}{2}\right)$$

$$-\ln(\cdot) = -\ln(\cdot)$$

$$-\ln p(S|X,h,\beta^{-1}) = -\frac{n}{2}\ln\beta + \frac{n}{2}\ln(2\pi) + \frac{\beta}{2}\sum_{i=1}^{n}(y_i - h(x_i))^2$$

$$= -\frac{n}{2}\ln\beta + \frac{n}{2}\ln(2\pi) + \frac{\beta}{2} n R_S(h)$$

Maximum A Posterior (MAP)

THE UNIVERSITY OF SYDNEY

Bayes' rule

$$p(\theta|S) = \frac{p(S|\theta)p(\theta)}{p(S)}$$

$$p(\theta|S) \propto p(S|\theta)p(\theta)$$

$$\arg\max_{\theta} p(\theta|S) = \arg\max_{\theta} p(S|\theta)p(\theta)$$

Maximum Posterior $P(\theta|S)$: the observed data makes the value of θ the "most probable".

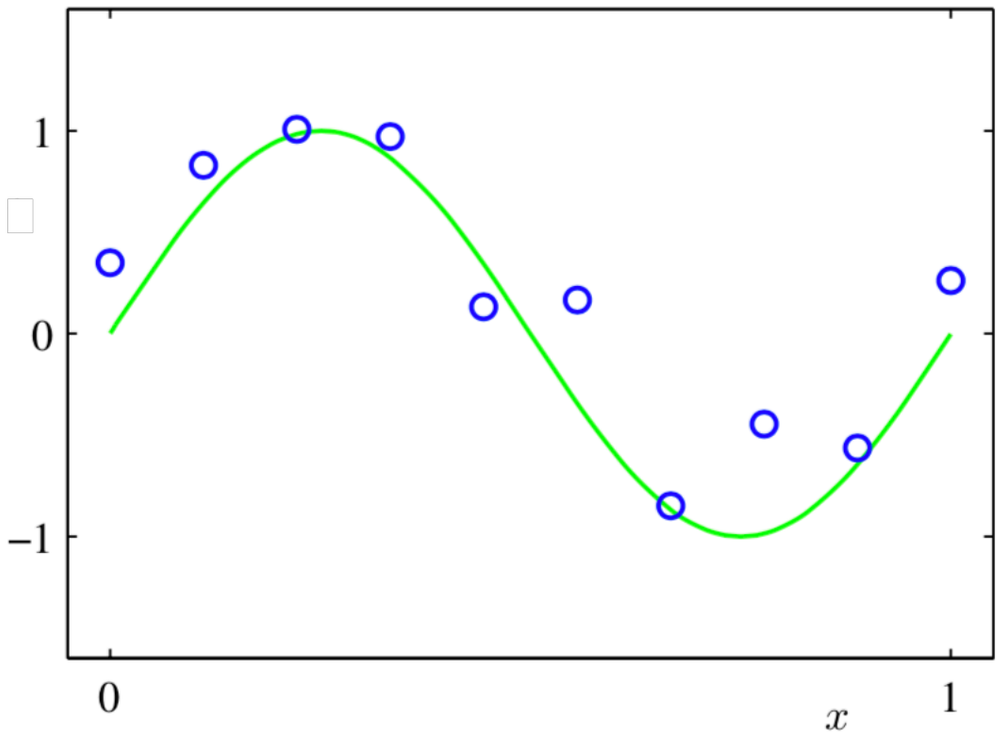
Maximum A Posterior (MAP)

Because of Bayes' rule

$$\arg\max_{\theta} p(\theta|S) = \arg\max_{\theta} p(S|\theta)p(\theta)$$

$$-\ln(\cdot) = -\ln(\cdot)$$

$$\begin{aligned}
\arg\min_{h}(-\ln p(h|S,\beta^{-1})) &= \arg\min_{h}(-\ln(p(S|X,h,\beta^{-1})p(h))) \\
&= \arg\min_{h}(-\ln p(S|X,h,\beta^{-1}) - \ln p(h)) \\
&= \arg\min_{h}(-\frac{n}{2}\ln\beta + \frac{n}{2}\ln(2\pi) + \frac{\beta}{2}nR_S(h) - \ln p(h))
\end{aligned}$$



$$h(x) = w_0 + w_1 x + \ldots + w_9 x^9$$

Assuming the prior distribution:

$$p(h) = \prod_{i=0}^{9} \sqrt{\frac{\tau}{2\pi}} \exp\left(-\frac{\tau w_i^2}{2}\right)$$

Then, we have

$$\arg\min_h(-\ln p(h|S, \beta^{-1})) = \arg\min_h(-\frac{n}{2}\ln\beta + \frac{n}{2}\ln(2\pi) + \frac{\beta}{2} n R_S(h)$$

$$- 5\ln\tau + 5\ln(2\pi) + \frac{\tau}{2} n \sum_{i=0}^{9} w_i^2)$$

Minimising above equals

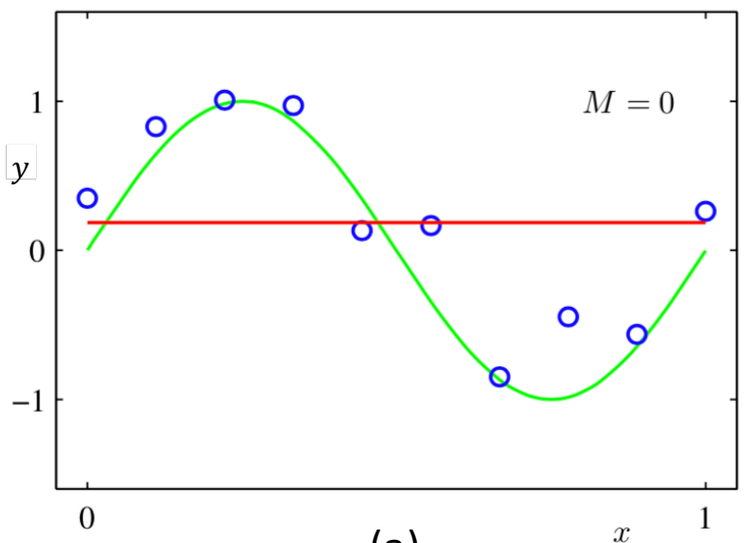
$$\min R_S(h) + \lambda \sum_{i=0}^{9} w_i^2 = R_S(h) + \boxed{\lambda \|w\|_2^2, \quad \lambda = \frac{\tau}{\beta}}$$

Bishop's book: "Pattern Recognition and Machine Learning"

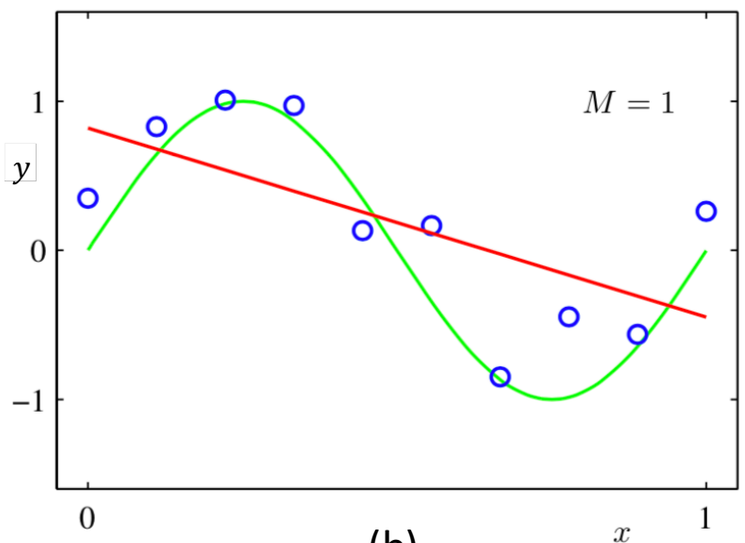
THE UNIVERSITY OF SYDNEY

Bias and variance

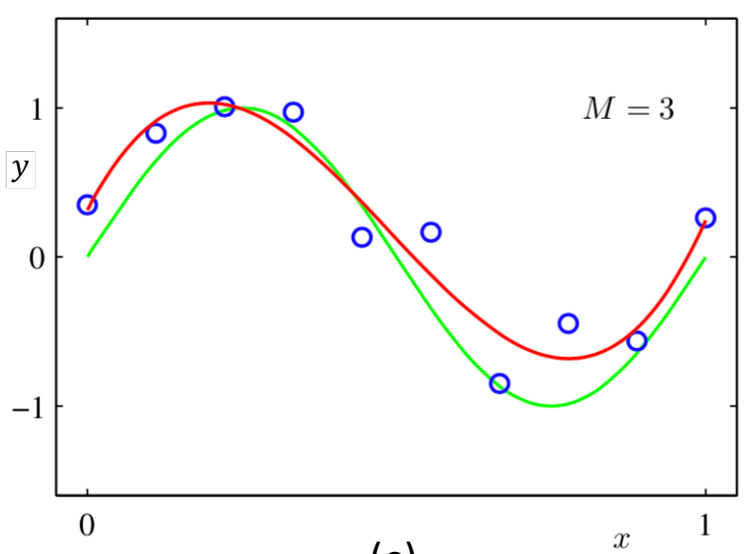
Linear regression



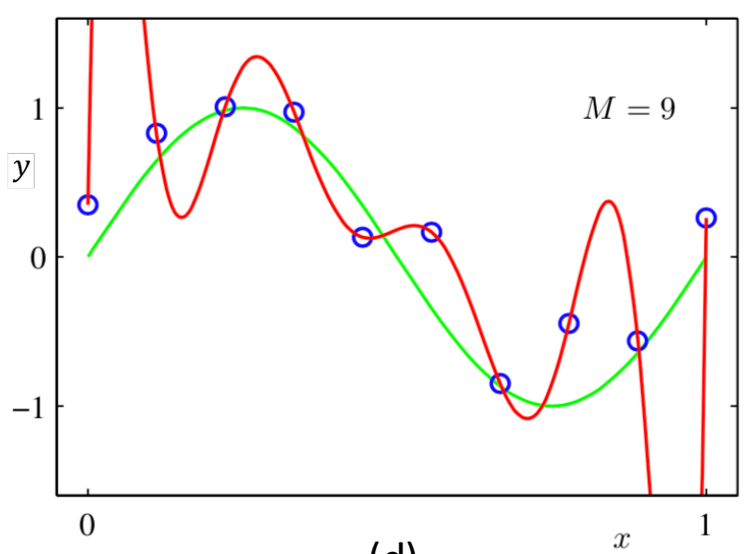
(a)



(b)



(c)



(d)

- True target: $\sin(2\pi x)$ with small Gaussian noises.
- $h(x) = w_0 + w_1 x + \cdots + w_M x^M$
- $R_S(h) = \frac{1}{n}\sum_{i=1}^{n}\big(y_i - h(x_i)\big)^2$

Bishop's book: "Pattern Recognition and Machine Learning"

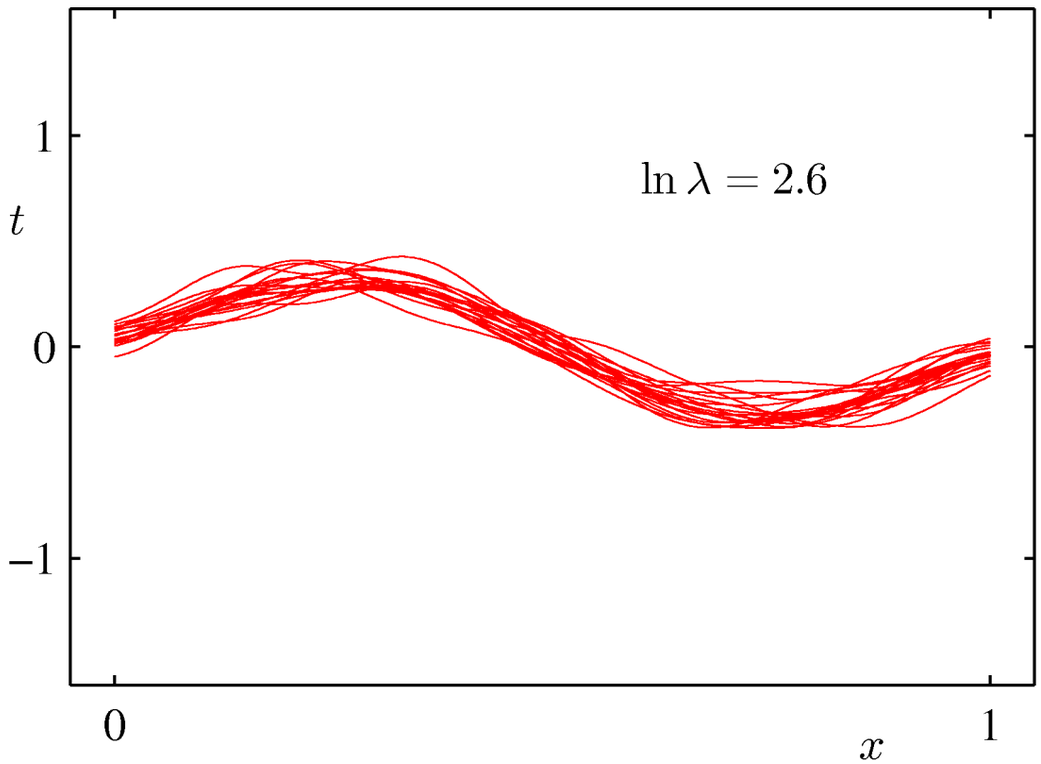
Underfitting and

- **Underfitting** is a phenomenon that the learned model does not fit the training data well; that is, large empirical risk.
- **Overfitting** is a phenomenon that the learned model fits the training data very well but it cannot generalise well to unseen examples drawn from the same distribution; that is, large difference between training and test errors.

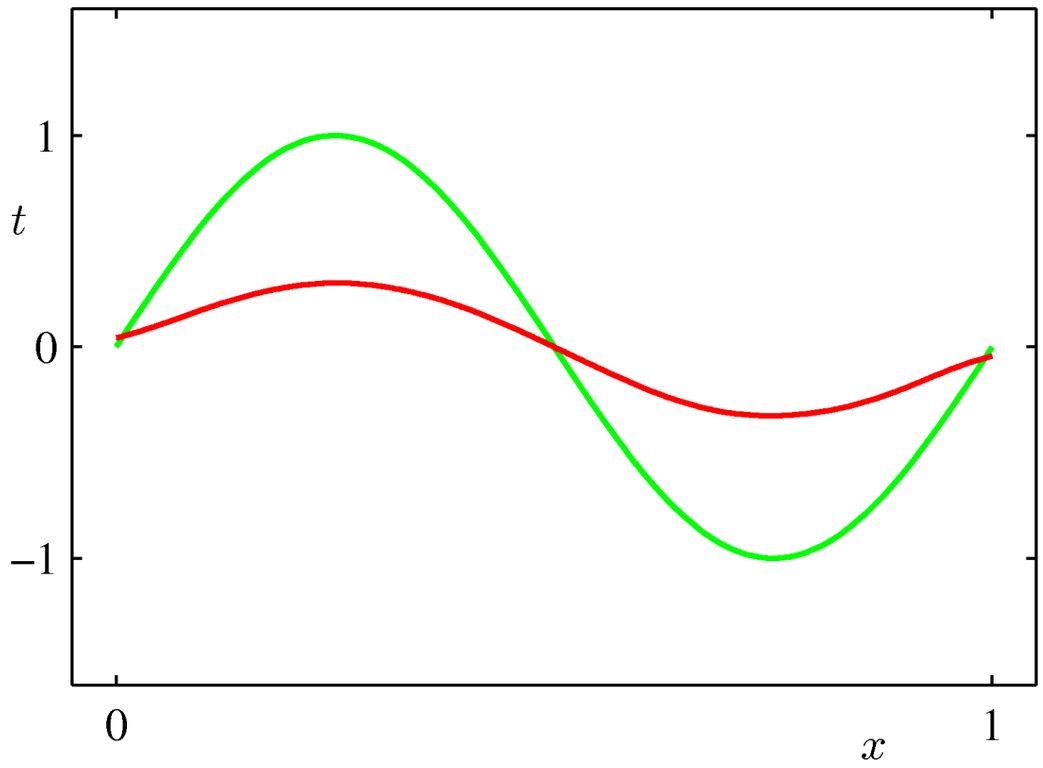
Bias-Variance Visualisation

THE UNIVERSITY OF SYDNEY

20 datasets with varying regularisation parameter.



Result of fitting the model to each dataset.

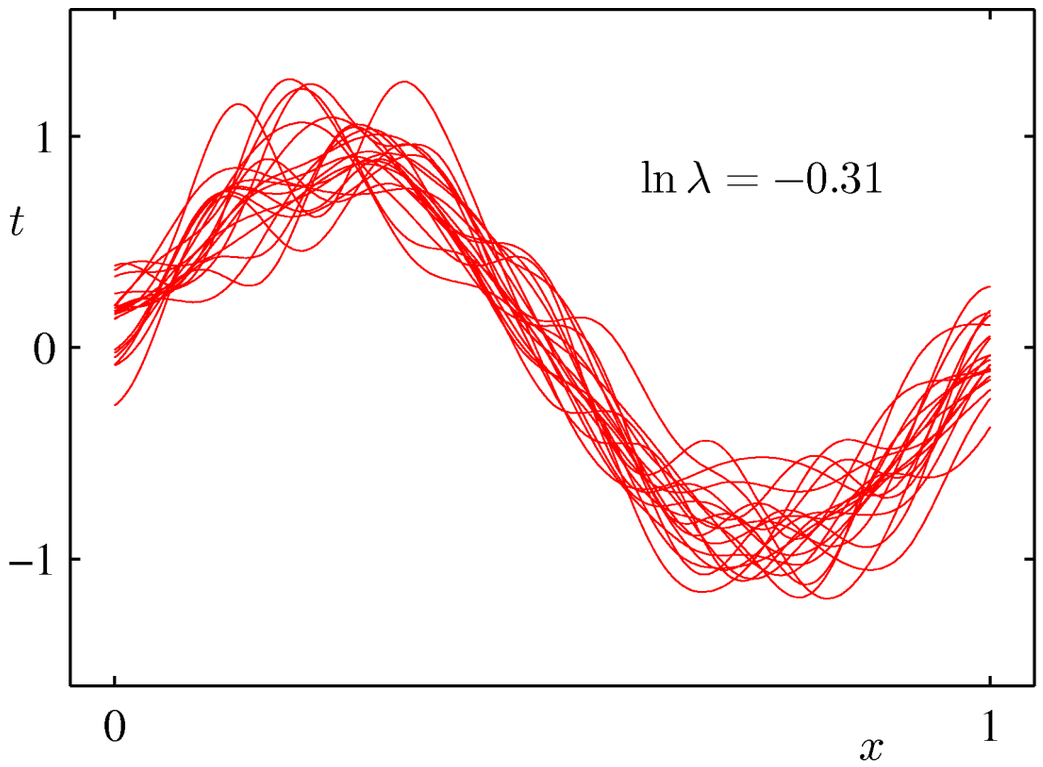


Average of the fits.

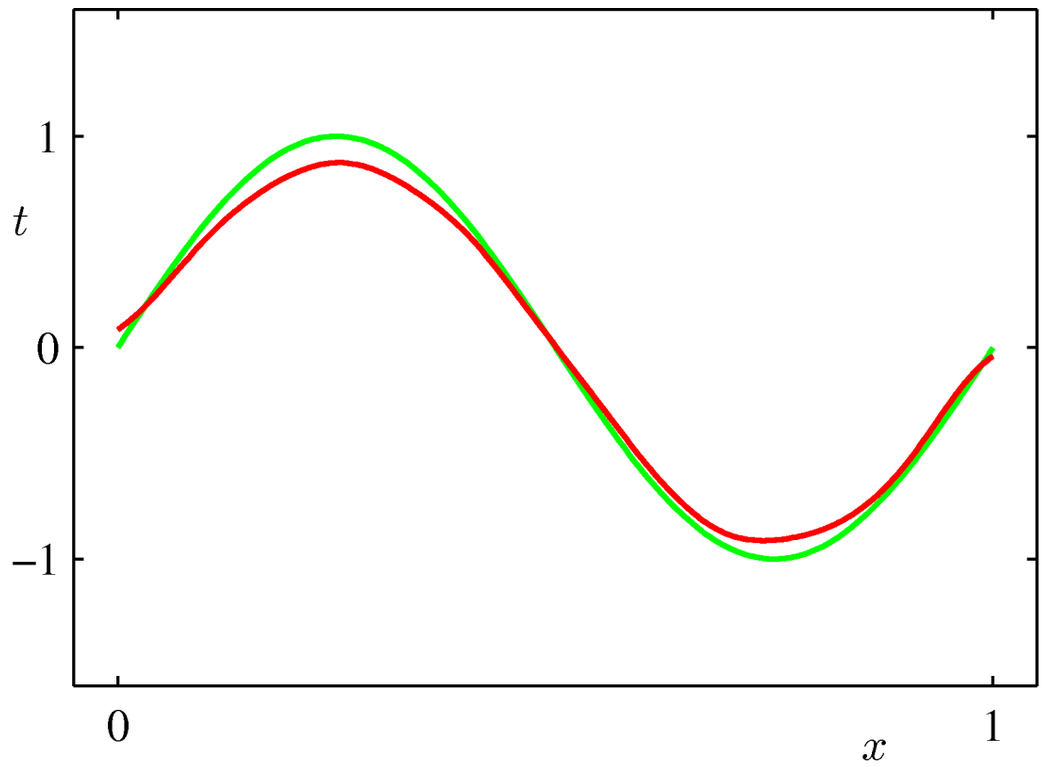
Bias-Variance Visualisation

THE UNIVERSITY OF SYDNEY

20 datasets with varying regularisation parameter.



Result of fitting the model to each dataset.

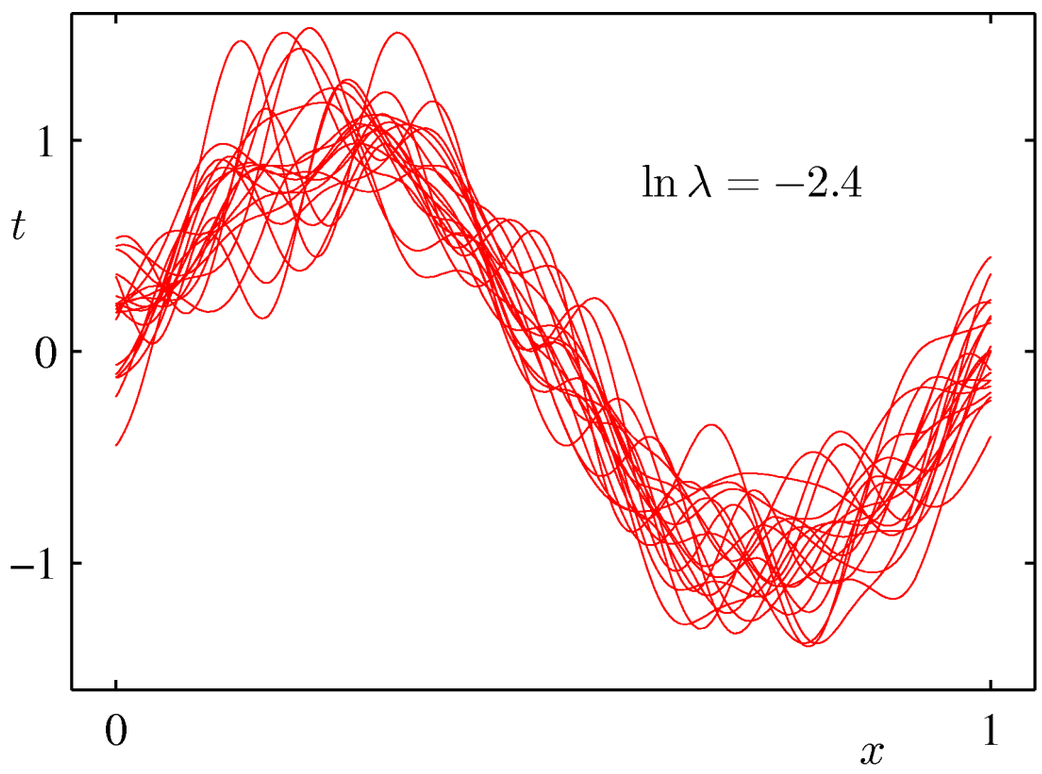


Average of the fits.

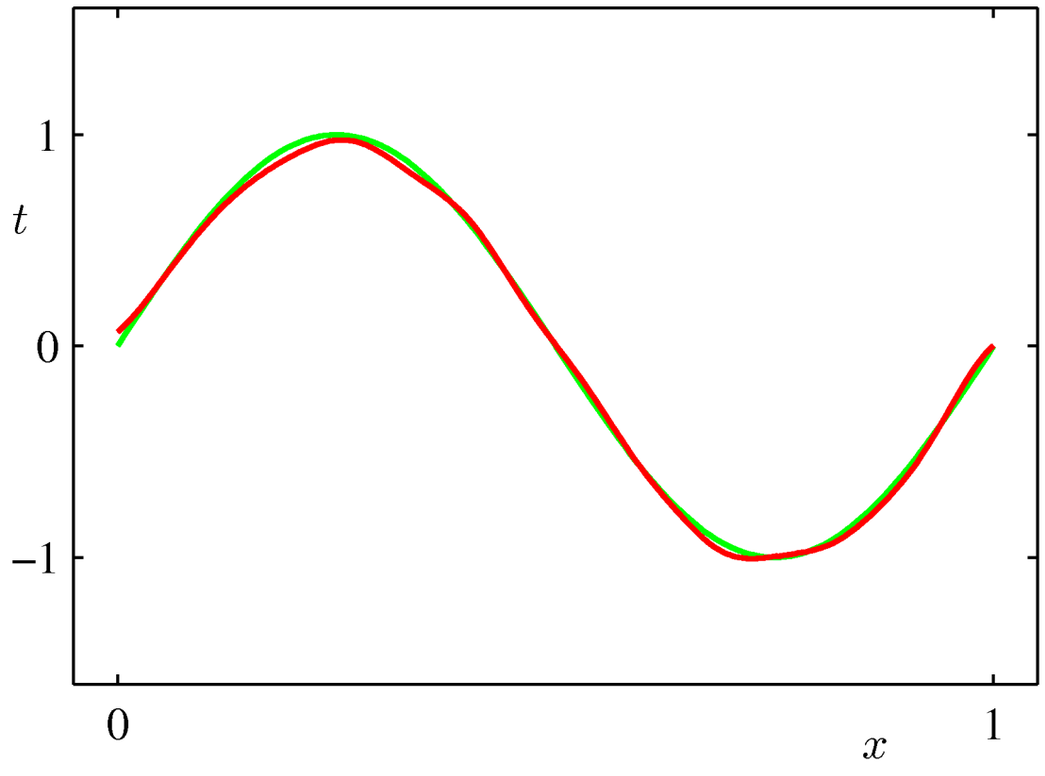
Bias-Variance Visualisation

THE UNIVERSITY OF SYDNEY

20 datasets with varying regularisation parameter.



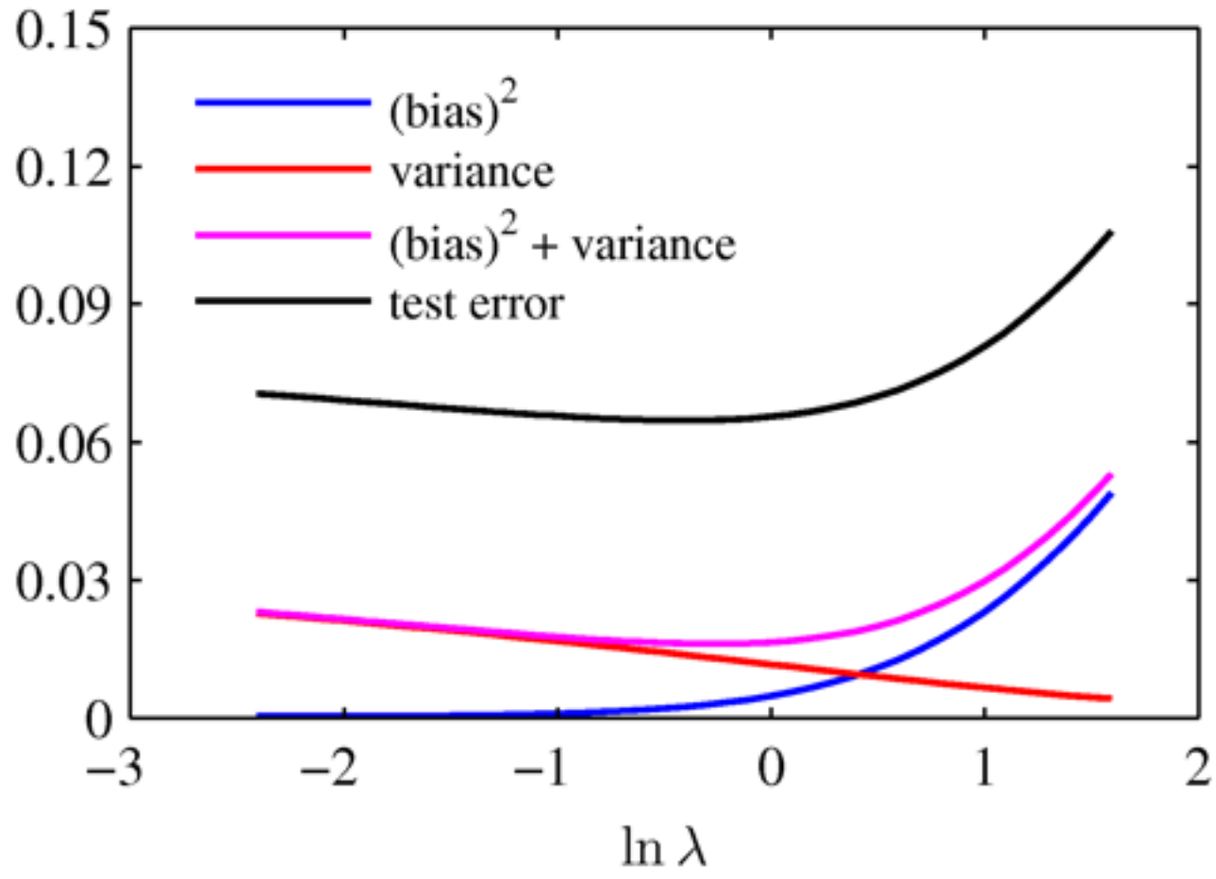
Result of fitting the model to each dataset.



Average of the fits.

The Bias-Variance Trade Off

From these plots, we note that an over-regularised model (large λ) will have a high bias, while an under-regularised model (small λ) will have a high variance.



Why the above phenomenon happens?

Bias-Variance vs Under-over fitting

- High variance implies overfitting.
- High bias implies underfitting.

Avoid overfitting

- Reducing the complexity of the predefined hypothesis class

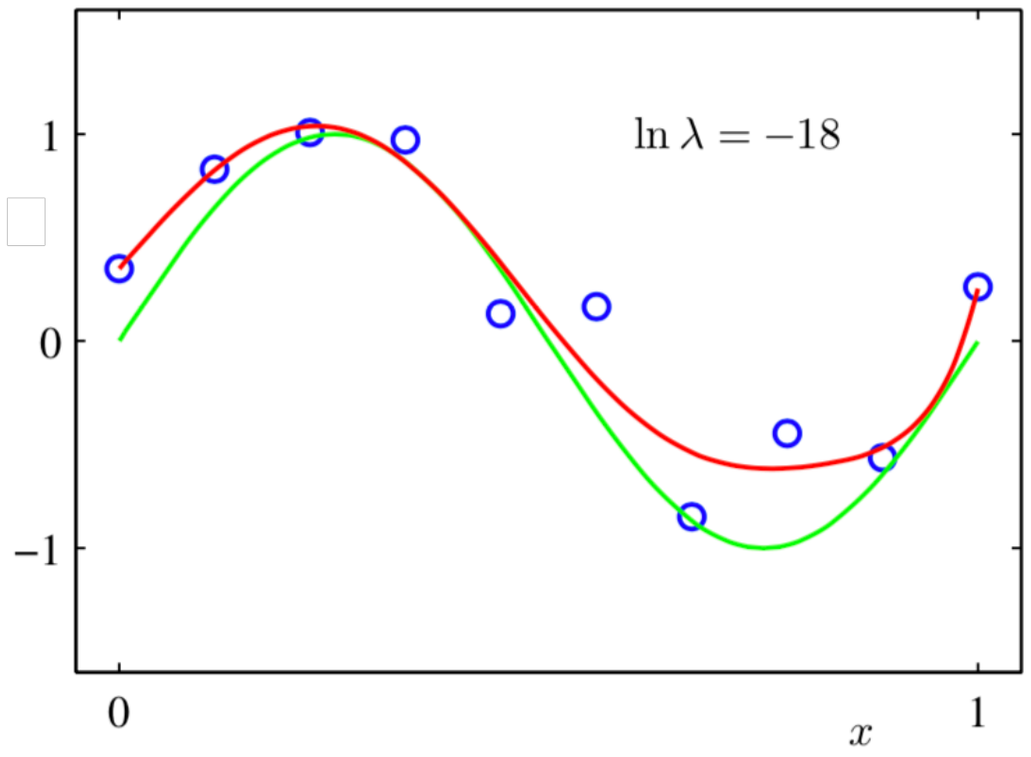
- Increasing training sample size

Why those two methods help?

Avoid overfitting

- **Reducing hypothesis complexity**: the hypothesis fits the training data too well because it is too complex.
- **Increasing sample size**: According to the law of large numbers, with more training examples, the empirical risk is closer to the expected risk. Increasing sample size will be helpful to learning the best hypothesis.

Example of regularised ERM



	$\ln \lambda = -\infty$	$\ln \lambda = -18$
$w_0^\star$	0.35	0.35
$w_1^\star$	232.37	4.74
$w_2^\star$	-5321.83	-0.77
$w_3^\star$	48568.31	-31.97
$w_4^\star$	-231639.30	-3.89
$w_5^\star$	640042.26	55.28
$w_6^\star$	-1061800.52	41.32
$w_7^\star$	1042400.18	-45.95
$w_8^\star$	-557682.99	-91.53
$w_9^\star$	125201.43	72.68

- $h(x) = w_0 + w_1 x + \cdots + w_9 x^9$
- $R_n(h) = \frac{1}{n}\sum_{i=1}^{n}\big(y_i - h(x_i)\big)^2 + \frac{\lambda}{2}\sum_{i=0}^{9} w_i^2.$
- $\|\mathrm{w}\|_2^2 = \sum_{i=0}^{9} w_i^2.$

Bishop's book: "Pattern Recognition and Machine Learning"

THE UNIVERSITY OF SYDNEY

Robustness of surrogate loss functions

Surrogate loss functions

- Least squares loss: $\ell(X, Y, h) = (Y - h(X))^2$
- Absolute loss: $\ell(X, Y, h) = |Y - h(X)|$
- Cauchy loss: $\ell(X, Y, h) = \log_2 \left(1 + \left(\frac{Y - h(X)}{\sigma}\right)^2\right)$
- Correntropy loss (Welsch loss):

$$\ell(X, Y, h) = \left(1 - \exp\left(-\left(\frac{Y - h(X)}{\sigma}\right)^2\right)\right)$$

Distribution of noise

THE UNIVERSITY OF SYDNEY

- Assumed noise: $\epsilon = Y - h(X)$

Gaussian distribution: $p(\epsilon|X, Y, h, \beta^{-1}) = \sqrt{\frac{\beta}{2\pi}} \exp\left(-\frac{\beta\epsilon^2}{2}\right)$

Laplacian distribution: $p(\epsilon|X, Y, h, \sigma) = \frac{1}{\sqrt{2}\sigma} \exp\left(-\frac{\sqrt{2}|\epsilon|}{\sigma}\right)$

Cauchy distribution: $p(\epsilon|X, Y, h, \gamma) = \frac{1}{\pi\gamma\left(1 + \left(\frac{\epsilon}{\gamma}\right)^2\right)}$

Laplacian regression (Least absolute deviation)

- Assumed noise: $\epsilon = Y - h(X)$

Laplacian distribution: $p(\epsilon|X, Y, h, \sigma) = \frac{1}{\sqrt{2}\sigma} \exp\left(-\frac{\sqrt{2}|\epsilon|}{\sigma}\right)$

Likelihood for one example:

$$p(y_i|x_i, h, b) = \frac{1}{\sqrt{2}\sigma} \exp\left(-\frac{\sqrt{2}|y_i - h(x_i)|}{\sigma}\right)$$

Laplacian regression (Least absolute deviation)

THE UNIVERSITY OF SYDNEY

- Assumed noise: $\epsilon = Y - h(X)$

Laplacian distribution: $p(\epsilon|X, Y, h, \sigma) = \frac{1}{\sqrt{2}\sigma} \exp\left(-\frac{\sqrt{2}|\epsilon|}{\sigma}\right)$

The likelihood function is

$$p(S|X, h, b) = \left(\frac{1}{\sqrt{2}\sigma}\right)^n \prod_{i=1}^{n} \exp\left(-\frac{\sqrt{2}|y_i - h(x_i)|}{\sigma}\right)$$

The negative log-likelihood function is

$$-\ln p(S|X, h, b) = n \ln(\sqrt{2}\sigma) + \frac{\sqrt{2}}{\sigma} \boxed{\sum_{i=1}^{n} |y_i - h(x_i)|}$$

Cauchy regression

- Assumed noise: $\epsilon = Y - h(X)$

Cauchy distribution: $p(\epsilon|X, Y, h, \gamma) = \dfrac{1}{\pi\gamma\left(1 + \left(\frac{\epsilon}{\gamma}\right)^2\right)}$

Likelihood for one example:

$$p(y_i|x_i, h, \gamma) = \frac{1}{\pi\gamma\left(1 + \left(\frac{y_i - h(x_i)}{\gamma}\right)^2\right)}$$

Cauchy regression

- Assumed noise: $\epsilon = Y - h(X)$

Cauchy distribution: $p(\epsilon|X, Y, h, \gamma) = \dfrac{1}{\pi\gamma\left(1 + \left(\dfrac{\epsilon}{\gamma}\right)^2\right)}$

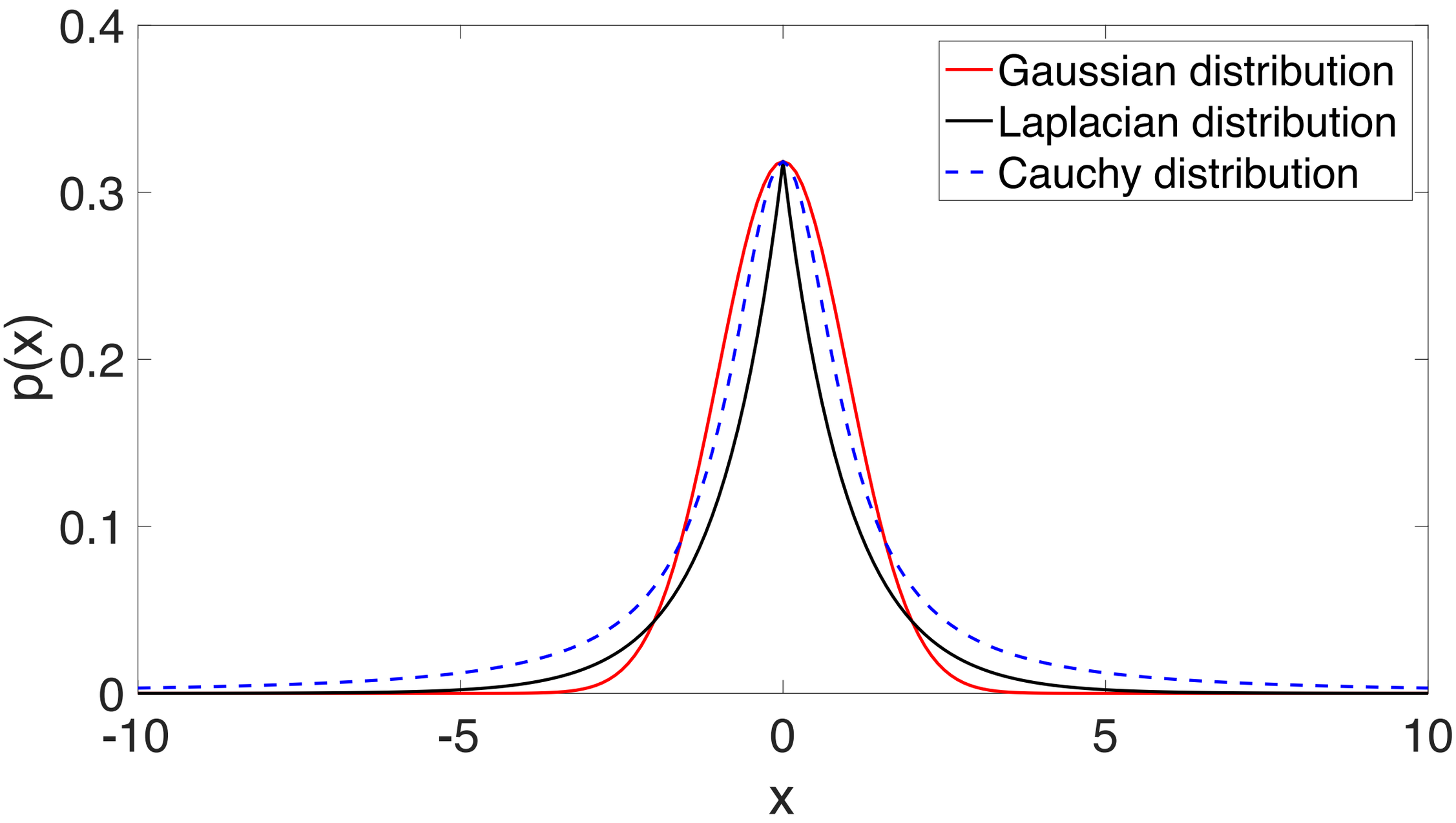
The likelihood function is

$$p(S|X, h, \gamma) = \left(\frac{1}{\pi\gamma}\right)^n \prod_{i=1}^{n} \frac{1}{1 + \left(\frac{y_i - h(x_i)}{\gamma}\right)^2}$$

The negative log-likelihood function is

$$-\ln p(S|X, h, \gamma) = n\ln(\pi\gamma) + \sum_{i=1}^{n} \ln\left(1 + \left(\frac{y_i - h(x_i)}{\gamma}\right)^2\right)$$

Distribution of noise



THE UNIVERSITY OF SYDNEY

Compare robustness among surrogate loss functions

Surrogate loss functions

- Least squares loss: $\ell(X, Y, h) = (Y - h(X))^2$
- Absolute loss: $\ell(X, Y, h) = |Y - h(X)|$
- Cauchy loss: $\ell(X, Y, h) = \ln\left(1 + \left(\frac{(Y - h(X))}{\gamma}\right)^2\right)$
- Correntropy loss (Welsch loss):

$$\ell(X, Y, h) = \left(1 - \exp\left(-\left(\frac{Y - h(X)}{\sigma}\right)^2\right)\right)$$

Objective function

Recall that a machine learning algorithm is a mapping to find a hypothesis to fit the data

$$\mathcal{A} : S \in (\mathcal{X} \times \mathcal{Y})^n \mapsto h_S \in H.$$

The mapping is an optimisation procedure that picks a hypothesis from the predefined hypothesis class to minimise or maximise the objective.

$$\arg\min_{h \in H} \frac{1}{n} \sum_{i=1}^{n} \ell(X_i, Y_i, h).$$

Optimality criterion

Let $$f(h) = \frac{1}{n}\sum_{i=1}^{n} \ell(X_i, Y_i, h).$$

A point *h* is optimal for f if and only if it is feasible and

$$\nabla f(h)^\top (h' - h) \geq 0$$

for all feasible $h' \in \text{domain } f$.

If the domain of *f* is not bounded, the optimality will be achieved at *h* such that

$$\nabla f(h) = \mathbf{0}.$$

Optimality criterion

Let

$$g(t) = f(th), t \in \mathbb{R}, h \in \mathbb{R}^d.$$

We have

$$g'(t) = \nabla f(th)^\top h.$$

If *h* is the minimiser, we have $\nabla f(h) = \mathbf{0}$. Then,

$$g'(1) = 0.$$

In other words, to minimise *f(h)*, we should find an *h* such that

$$g'(1) = 0.$$

Surrogate loss function robustness

THE UNIVERSITY OF SYDNEY

- Least squares loss: $\ell(X, Y, h) = (Y - h(X))^2$

$$g'(1) = \frac{1}{n}\sum_{i=1}^{n} 2(y_i - h(x_i))(-h(x_i))$$

- Absolute loss: $\ell(X, Y, h) = |Y - h(X)|$

$$g'(1) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{|y_i - h(x_i)|}(y_i - h(x_i))(-h(x_i))$$

We define the derivative of $|x|$ at 0 can be any real value in $[-1, 1]$.

Liu, Tongliang, and Dacheng Tao. "On the robustness and generalization of Cauchy regression." Information Science and Technology (ICIST), 2014 4th IEEE International Conference on. IEEE, 2014.

Surrogate loss function robustness

THE UNIVERSITY OF SYDNEY

- Cauchy loss: $\ell(X,Y,h) = \ln\left(1+\left(\frac{(Y-h(X))}{\gamma}\right)^2\right)$

$$g'(1) = \frac{1}{n}\sum_{i=1}^{n}\frac{2}{\gamma^2+(y_i-h(x_i))^2}(y_i-h(x_i))(-h(x_i))$$

- Correntropy loss (Welsch loss):

$$\ell(X,Y,h) = \left(1-\exp\left(-\left(\frac{Y-h(X)}{\sigma}\right)^2\right)\right)$$

$$g'(1) = \frac{1}{n}\sum_{i=1}^{n}\frac{2}{\sigma^2\exp\left(\frac{y_i-h(x_i)}{\sigma}\right)^2}(y_i-h(x_i))(-h(x_i))$$

Liu, Tongliang, and Dacheng Tao. "On the robustness and generalization of Cauchy regression." Information Science and Technology (ICIST), 2014 4th IEEE International Conference on. IEEE, 2014.

Surrogate loss function robustness

THE UNIVERSITY OF SYDNEY

- Least squares loss: $g'(1) = \frac{1}{n}\sum_{i=1}^{n} 2(y_i - h(x_i))(-h(x_i))$
- Absolute loss: $g'(1) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{|y_i - h(x_i)|}(y_i - h(x_i))(-h(x_i))$
- Cauchy loss:

$$g'(1) = \frac{1}{n}\sum_{i=1}^{n} \frac{2}{\gamma^2 + (y_i - h(x_i))^2}(y_i - h(x_i))(-h(x_i))$$

- Correntropy loss (Welsch loss):

$$g'(1) = \frac{1}{n}\sum_{i=1}^{n} \frac{2}{\sigma^2 \exp\left(\frac{y_i - h(x_i)}{\sigma}\right)^2}(y_i - h(x_i))(-h(x_i))$$

Surrogate loss function

THE UNIVERSITY OF SYDNEY

In other words, to minimise *f(h)*, we should find an *h* such that

$$g'(1) = 0.$$

- All $g'(1)$ w.r.t. the above four loss functions has the term $c_i = (y_i - h(x_i))(-h(x_i))$, we treat them as the bases of contribution to optimising the empirical risks. We can see that different surrogate loss functions assign different weights to the bases. A surrogate loss function is more robust if it assign a smaller weights to the bases as the error (or noise) is going bigger.

Surrogate loss function

THE UNIVERSITY OF SYDNEY

For a given task, we have some training data, but don't know the quality of the data. What's kind of loss function you will chose to design a learning algorithm?

THE UNIVERSITY OF SYDNEY

Research

NMF robustness

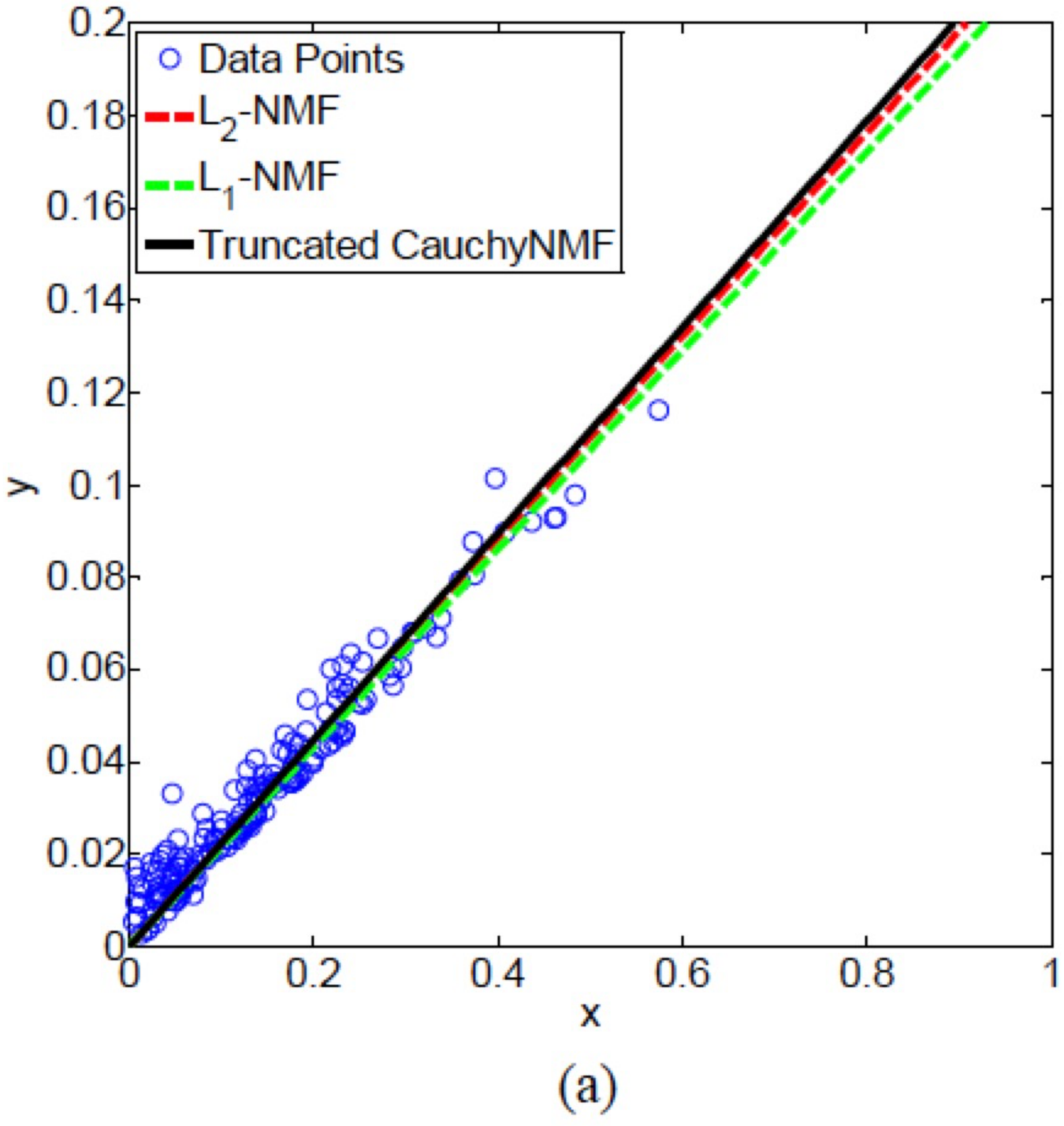
Non-negative matrix factorisation

$$\min_{D \in \mathcal{D}, R \in \mathcal{R}} \|X - DR\|_F^2$$

Special requirement: $\mathcal{D} = \mathbb{R}_+^{d \times k}, \quad \mathcal{R} = \mathbb{R}_+^{k \times n}.$

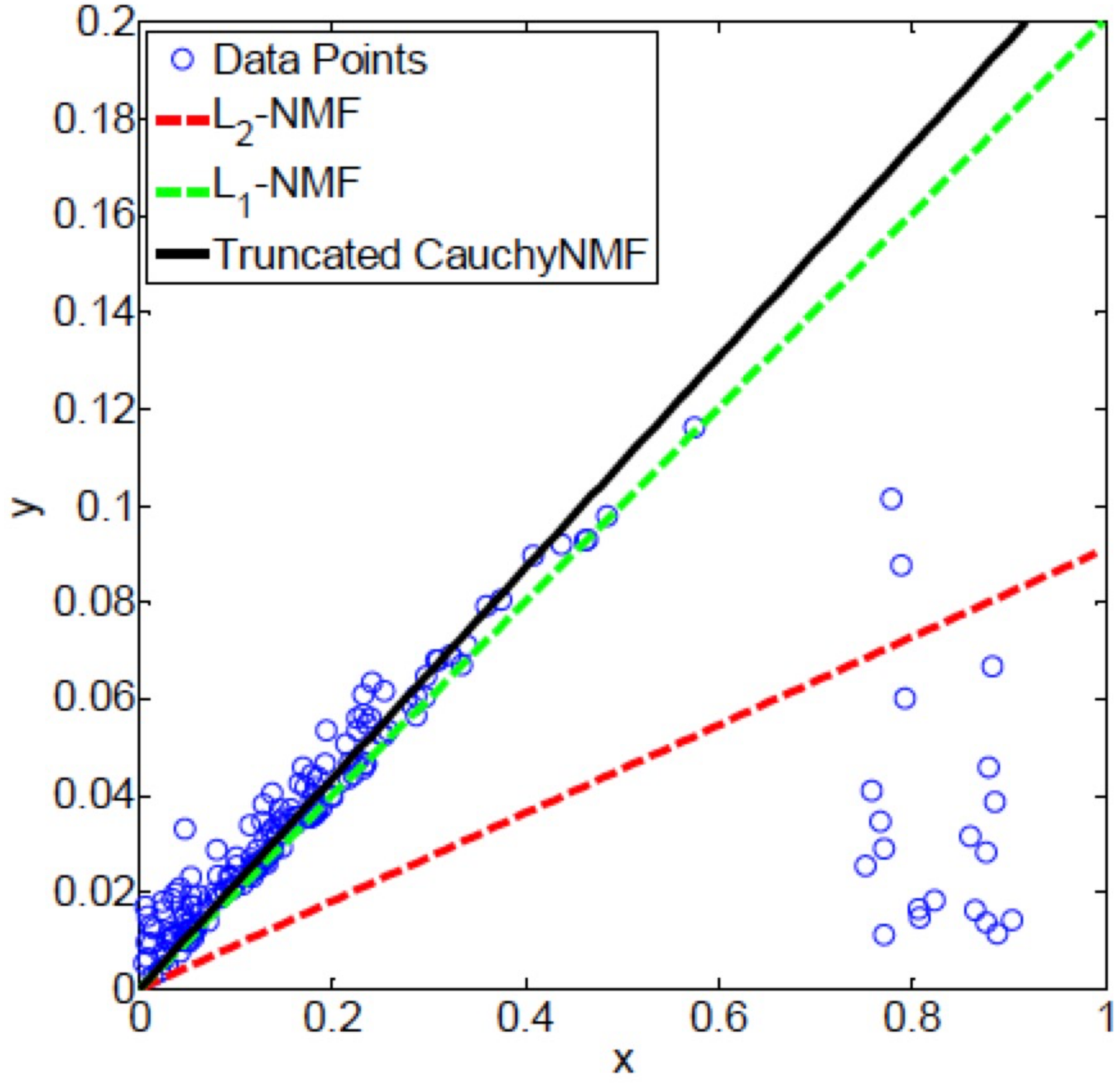
Employ more robust surrogate loss functions other than least squares

$$\min_{D \in \mathcal{D}, R \in \mathcal{R}} \sum_{i=1}^{n} \ell(X_{:,i} - DR_{:,i})$$



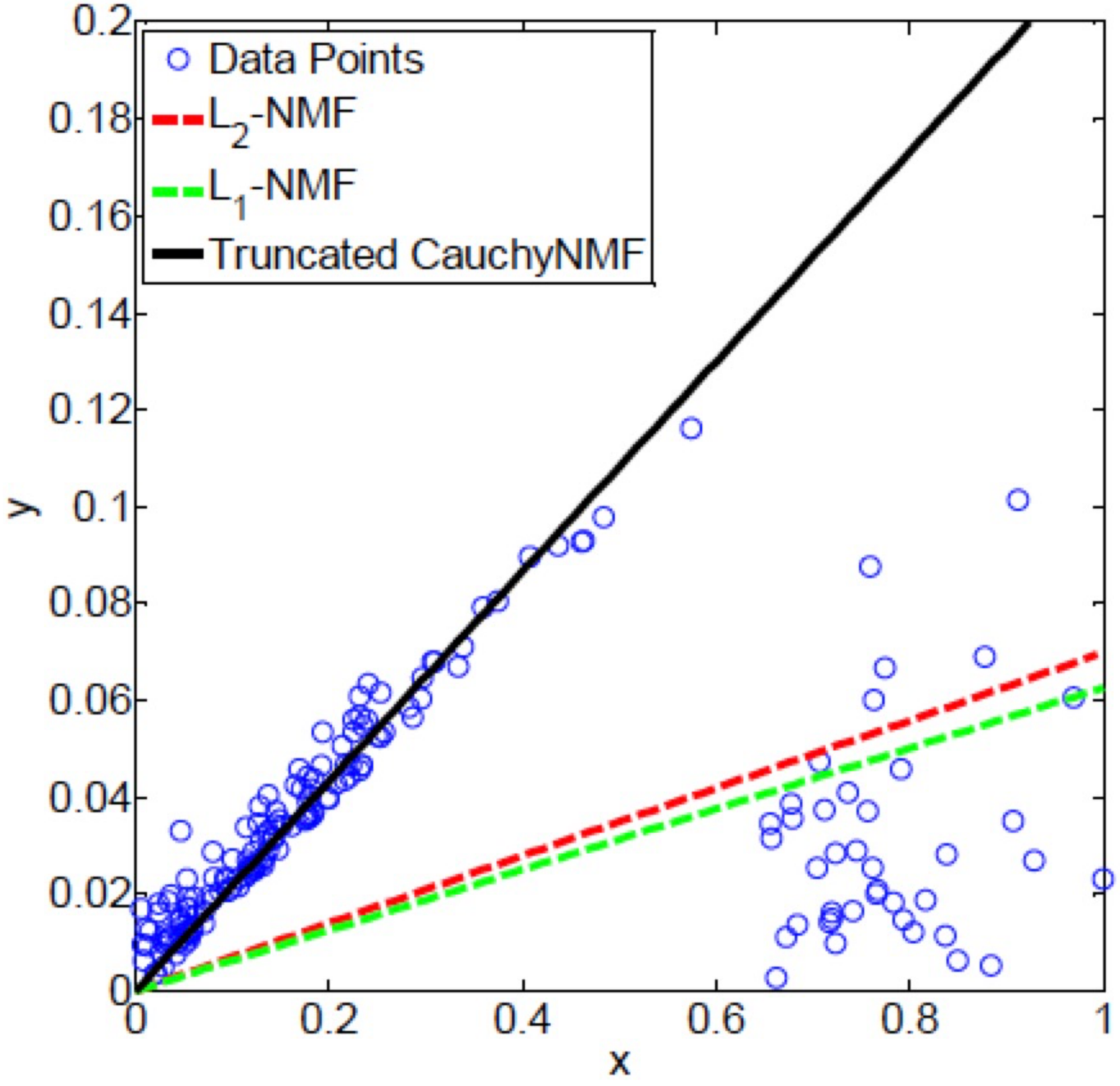
(a)

Guan, Naiyang, et al. "Truncated Cauchy Non-negative Matrix Factorization for Robust Subspace Learning." IEEE Transactions on Pattern Analysis and Machine Intelligence (2017).



(b)

Guan, Naiyang, et al. "Truncated Cauchy Non-negative Matrix Factorization for Robust Subspace Learning." IEEE Transactions on Pattern Analysis and Machine Intelligence (2017).



(c)

Guan, Naiyang, et al. "Truncated Cauchy Non-negative Matrix Factorization for Robust Subspace Learning." IEEE Transactions on Pattern Analysis and Machine Intelligence (2017).