Neural Network Estimation of Likelihood Ratios

for Testing and Detection

George V. Moustakides
University of Patras, Greece
Rutgers University, USA

Kalliopi Basioti
Rutgers University, USA

Outline

- Problem definition
- Optimization problems with pre-specified solutions
- Examples
- Data-driven version
- Applications
  - Hypothesis testing and classification
  - Generalized likelihood ratio test
  - CUSUM (sequential change detection)

In Hypothesis Testing and Detection, every time a data vector X is acquired, we like to decide between

$$\begin{aligned} \mathsf{H}_0 &: \quad X \sim \mathsf{f}_0(X) \\ \mathsf{H}_1 &: \quad X \sim \mathsf{f}_1(X) \end{aligned}$$

The optimum test is the **Likelihood Ratio Test** which can come under different forms

$$\frac{\mathsf{f}_1(X)}{\mathsf{f}_0(X)} \underset{\mathsf{H}_0}{\overset{\mathsf{H}_1}{\gtreqless}} \nu, \qquad \log\frac{\mathsf{f}_1(X)}{\mathsf{f}_0(X)} \underset{\mathsf{H}_0}{\overset{\mathsf{H}_1}{\gtreqless}} \nu', \qquad \mathsf{p}_1(X) \underset{\mathsf{H}_0}{\overset{\mathsf{H}_1}{\gtreqless}} \mathsf{p}_0(X)$$

Bayesian

Posterior probability

$$\mathsf{p}_1(X) = \frac{\frac{\mathsf{f}_1(X)}{\mathsf{f}_0(X)}}{1 + \frac{\mathsf{f}_1(X)}{\mathsf{f}_0(X)}}$$

$$\mathsf{H}_0: \quad X \sim \mathsf{f}_0(X)$$
$$\mathsf{H}_1: \quad X \sim \mathsf{f}_1(X)$$

Knowledge of $\mathsf{f}_0(X), \mathsf{f}_1(X)$

Can we replace the need for knowing the two densities **with the requirement to have available data sampled from the two densities ?**

$$\mathsf{H}_0: \quad X \sim \cancel{\mathsf{f}_0(X)} \qquad \{X_1^0, X_2^0, \ldots, X_{n_0}^0\}$$
$$\mathsf{H}_1: \quad X \sim \cancel{\mathsf{f}_1(X)} \qquad \{X_1^1, X_2^1, \ldots, X_{n_1}^1\}$$

Can we estimate

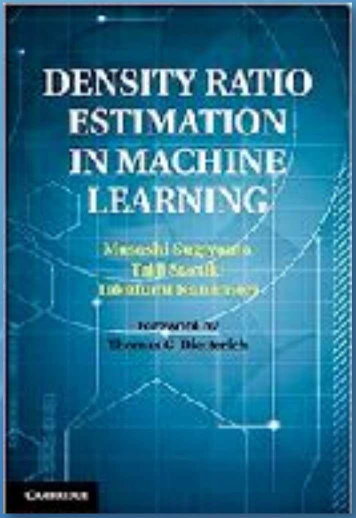
$$\frac{\mathsf{f}_1(X)}{\mathsf{f}_0(X)}, \quad \log\frac{\mathsf{f}_1(X)}{\mathsf{f}_0(X)}, \quad \frac{\frac{\mathsf{f}_1(X)}{\mathsf{f}_0(X)}}{1+\frac{\mathsf{f}_1(X)}{\mathsf{f}_0(X)}} \quad \ldots$$

using Neural Networks ?

M. Sugiyama, T. Suzuki, T. Kanamori, Density ratio estimation: A comprehensive review'', *RIMS Kokyuroku*, vol. 1703, pp. 10-31, 2010.

.....

M. Sugiyama, T. Suzuki, T. Kanamori, *Density Ratio Estimation in Machine Learning*, Cambridge, 2013.



- Focus only on density ratio estimation (likelihood ratio function) with SVM
- No estimates of nonlinear transformations, like log-likelihood ratio or posterior probability
- Difficult to include positivity condition

We are interested in estimates with Neural Networks

Training of Neural Networks: **Optimization Problems**

General Optimization Problem

Let X random and follows $\mathsf{f}_0(X)$. Consider the cost

$$\mathcal{J}(\mathsf{u}) = \mathsf{E}_0\left[\phi\big(\mathsf{u}(X)\big) + \mathsf{r}(X)\psi\big(\mathsf{u}(X)\big)\right]$$

$\mathsf{u}(X)$, $\mathsf{r}(X)$ scalar functions of X.
$\phi(z)$, $\psi(z)$ scalar functions of scalar z.
$\omega(\mathsf{r})$ scalar functions of scalar r.

Interested in

$$\min_{\mathsf{u}(X)} \mathcal{J}(\mathsf{u}) = \min_{\mathsf{u}(X)} \mathsf{E}_0\left[\phi\big(\mathsf{u}(X)\big) + \mathsf{r}(X)\psi\big(\mathsf{u}(X)\big)\right]$$

Design $\phi(z), \psi(z)$, s.t. $\mathsf{u}_o(X) = \omega\big(\mathsf{r}(X)\big)$

Theorem

If $\omega(\mathrm{r})$ known scalar function of r, for the minimizer to satisfy $\mathrm{u}_o(X) = \omega(\mathrm{r}(X))$, **necessary condition**:

$$\phi'\big(\omega(\mathrm{r})\big) + \mathrm{r}\psi'\big(\omega(\mathrm{r})\big) = 0, \quad \forall \mathrm{r} \in \mathrm{I}_\mathrm{r}$$

where I_r the range of $\mathrm{r}(X)$

No $\mathrm{f}_0(X)$ and $\mathrm{r}(X)$

If $\omega(\mathrm{r})$ strictly increasing, then equivalently

$$\phi'(z) + \omega^{-1}(z)\psi'(z) = 0, \quad \forall z \in \omega(\mathrm{I}_\mathrm{r})$$

where $\omega^{-1}(z)$ the **inverse function** of $\omega(\mathrm{r})$, and $\omega(\mathrm{I}_\mathrm{r})$ the image of I_r under $\omega(\mathrm{r})$.

Theorem (cont.)

If $\rho(z) < 0$ and we define $\forall z \in \omega(\mathsf{I}_r)$

$$\phi'(z) = -\omega^{-1}(z)\rho(z)$$
$$\psi'(z) = \rho(z),$$

No knowledge of $f_0(X)$ or $r(X)$ required !!!

then

$$\mathcal{J}(u) = E_0\big[\phi\big(u(X)\big) + r(X)\psi\big(u(X)\big)\big]$$

has a **single extremum** which is equal to

$$u_o(X) = \omega\big(r(X)\big)$$

and this extremum is a **minimum**.

Case $\omega(\mathrm{r}) = \mathrm{r} > 0$

$$\rho(z) = -z^{\alpha} \Rightarrow \begin{cases} \phi(z) = \frac{z^{2+\alpha}}{2+\alpha} \\ \psi(z) = -\frac{z^{1+\alpha}}{1+\alpha} \end{cases}; \quad z \in (0, \infty)$$

$$\min_{\mathrm{u}(X)} \mathcal{J}(\mathrm{u}) = \min_{\mathrm{u}(X)} \mathsf{E}_0\big[\phi(\mathrm{u}(X)) + \mathrm{r}(X)\psi(\mathrm{u}(X))\big]$$

$$\Downarrow$$

$$\mathrm{u}_o(X) = \mathrm{r}(X)$$

Most popular case: $\alpha = 0$. Criterion is **Mean Square Error**

$$\mathcal{J}(\mathrm{u}) = \frac{1}{2}\mathsf{E}_0\big[(\mathrm{u}(X) - \mathrm{r}(X))^2\big] + C$$

Case $\omega(\mathrm{r}) = \log \mathrm{r}$

$$\rho(z) = -e^{-\alpha z} \Rightarrow \begin{cases} \phi(z) = \dfrac{e^{(1-\alpha)z}}{1-\alpha} \\ \psi(z) = \dfrac{e^{-\alpha z}}{\alpha} \end{cases}, \quad z \in \mathbb{R}$$

Exponential loss: $\alpha = 0.5$, $\phi(z) = e^{0.5z}$, $\psi(z) = e^{-0.5z}$

$$\min_{\mathrm{u}(X)} \mathcal{J}(\mathrm{u}) = \min_{\mathrm{u}(X)} \mathrm{E}_0\big[\phi(\mathrm{u}(X)) + \mathrm{r}(X)\psi(\mathrm{u}(X))\big]$$

$$\Downarrow$$

$$\mathrm{u}_o(X) = \log\big(\mathrm{r}(X)\big)$$

Case $\omega(\mathrm{r}) = \dfrac{\mathrm{r}}{1+\mathrm{r}}$

$$\rho(z) = -\frac{1}{z} \;\Rightarrow\; \begin{cases} \phi(z) = -\log(1-z) \\ \psi(z) = -\log z \end{cases}, \quad z \in (0,1)$$

Criterion known as **Cross-Entropy loss**

Case $\omega(\mathrm{r}) = \mathrm{sign}(\log \mathrm{r})$

$$\cdot(z) = -\mathbb{1}_{\{z < -1\}}\ ) \qquad \begin{array}{l} \phi(z) = \max\{1+z, 0\} \\ \ (z) = \max\{1-z, 0\} \end{array}, \ z \, 2 \, \mathrm{R}$$

Criterion known as **Hinge loss**

$$\rho(z) = -1 \;\Rightarrow\; \phi(z) = z, \quad \psi(z) = -z, \quad z \in [-1, 1]$$

Linear loss

$$\min_{\mathsf{u}(X)} \mathcal{J}(\mathsf{u}) = \min_{\mathsf{u}(X)} \mathsf{E}_0\left[\phi(\mathsf{u}(X)) + \mathsf{r}(X)\psi(\mathsf{u}(X))\right]$$

$$\mathsf{r}(X) = \frac{\mathsf{f}_1(X)}{\mathsf{f}_0(X)}$$

$$\mathcal{J}(\mathsf{u}) = \mathsf{E}_0[\phi(\mathsf{u}(X))] + \mathsf{E}_1[\psi(\mathsf{u}(X))]$$

$$\Big\downarrow \text{ Data}$$

$$\mathcal{J}(\mathsf{u}) \approx \frac{1}{n_0}\sum_{i=1}^{n_0}\phi(\mathsf{u}(X_i^0)) + \frac{1}{n_1}\sum_{i=1}^{n_1}\psi(\mathsf{u}(X_i^1))$$

Limit $\mathsf{u}(X)$ to a Neural Network output $\mathsf{u}(X,\theta)$

$$\mathcal{J}(\mathsf{u}) \approx \hat{\mathcal{J}}(\theta) = \frac{1}{n_0}\sum_{i=1}^{n_0}\phi(\mathsf{u}(X_i^0,\theta)) + \frac{1}{n_1}\sum_{i=1}^{n_1}\psi(\mathsf{u}(X_i^1,\theta))$$

$$\min_{\mathsf{u}(X)} \mathcal{J}(\mathsf{u}) \approx \min_{\theta} \hat{\mathcal{J}}(\theta)$$

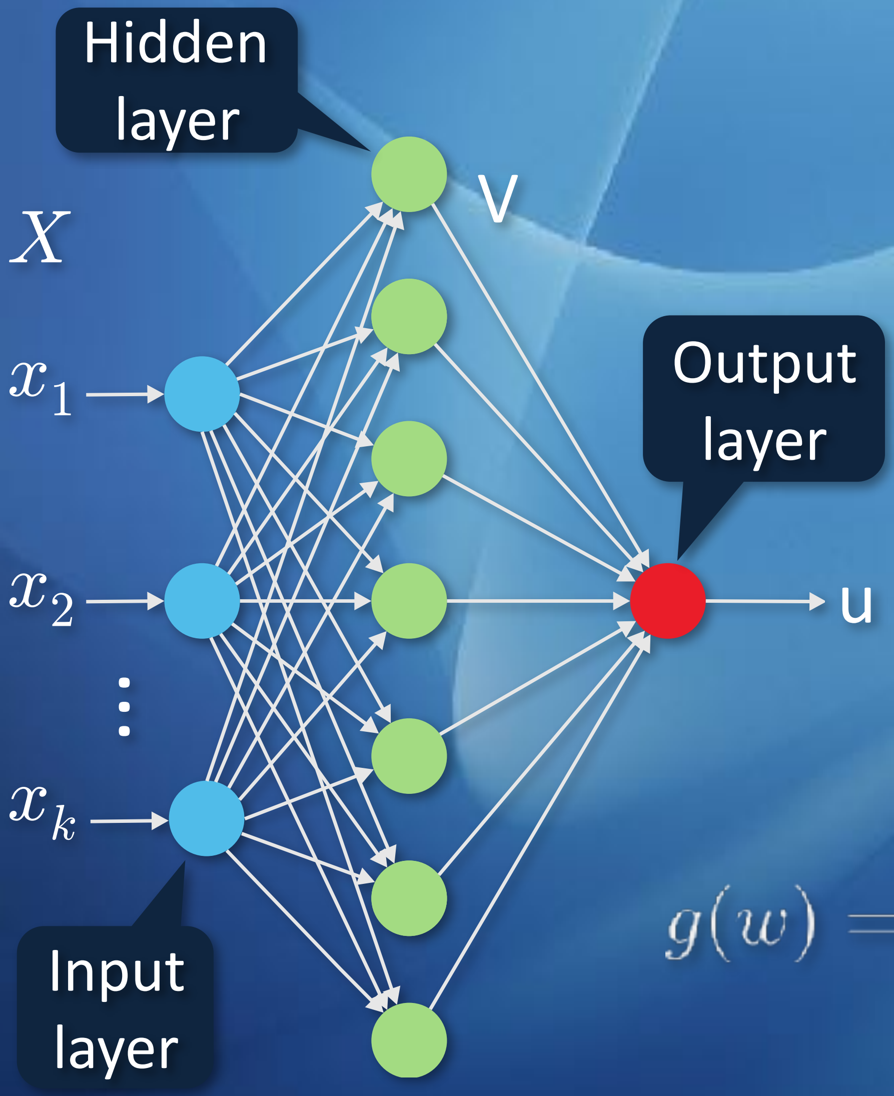
$$= \min_{\theta} \left\{ \frac{1}{n_0} \sum_{i=1}^{n_0} \phi(\mathsf{u}(X_i^0, \theta)) + \frac{1}{n_1} \sum_{i=1}^{n_1} \psi(\mathsf{u}(X_i^1, \theta)) \right\}$$

$$\hookrightarrow \theta_o \longrightarrow \mathsf{u}(X, \theta_o) \approx \mathsf{u}_o(X)$$

Depending on the selected $\phi(z)$, $\psi(z)$ the neural network $\mathsf{u}(X,\theta_o)$ can approximate

$$\frac{\mathsf{f}_1(X)}{\mathsf{f}_1(X)}, \ \log\frac{\mathsf{f}_1(X)}{\mathsf{f}_1(X)}, \ \frac{\mathsf{f}_1(X)}{\mathsf{f}_1(X) + \mathsf{f}_0(X)}, \ \mathsf{sign}\left(\log\frac{\mathsf{f}_1(X)}{\mathsf{f}_1(X)}\right)$$

without any knowledge of the two densities



$$U = AX + a$$

$$V = \max\{U, 0\}$$

ReLU

$$w = B^{\intercal} V + b$$

$$u = g(w)$$

Activation function

$$\theta = \{A, a, B, b\}$$

$$g(w) = \begin{cases} \max\{w, 0\}, & \text{mean square} \\ w, & \text{exponential} \\ \frac{1}{1+e^{-w}}, & \text{cross-entropy} \\ \tanh w, & \text{linear} \\ w, & \text{hinge} \end{cases}$$

$$\min_{\theta}\left\{\frac{1}{n_0}\sum_{i=1}^{n_0}\phi\big(\mathsf{u}(X_i^0,\theta)\big)+\frac{1}{n_1}\sum_{i=1}^{n_1}\psi\big(\mathsf{u}(X_i^1,\theta)\big)\right\}$$

Gradient Descent

$$\theta_t=\theta_{t-1}-$$
$$\mu\left\{\frac{1}{n_0}\sum_{i=1}^{n_0}\nabla_\theta\phi\big(\mathsf{u}(X_i^0,\theta_{t-1})\big)+\frac{1}{n_1}\sum_{i=1}^{n_1}\nabla_\theta\psi\big(\mathsf{u}(X_i^1,\theta_{t-1})\big)\right\}$$

In every iteration all data

If $n_0 = n_1$ then Stochastic Gradient Descent

$$\theta_t=\theta_{t-1}-\mu\left\{\nabla_\theta\phi\big(\mathsf{u}(X_t^0,\theta_{t-1})\big)+\nabla_\theta\psi\big(\mathsf{u}(X_t^1,\theta_{t-1})\big)\right\}$$

In every iteration one pair of data

Hypothesis testing

$$\mathsf{H}_0 : \quad X \sim \cancel{\mathsf{f}_0(X)} \qquad \{X_1^0, X_2^0, \ldots, X_{n_0}^0\}$$

$$\mathsf{H}_1 : \quad X \sim \cancel{\mathsf{f}_1(X)} \qquad \{X_1^1, X_2^1, \ldots, X_{n_1}^1\}$$

$$X \in \mathbb{R}^{10}$$

$$\mathsf{f}_0(X) = \mathcal{N}(\mu_0, \Sigma_0) \qquad \mathsf{f}_1(X) = \mathcal{N}(\mu_1, \Sigma_1)$$

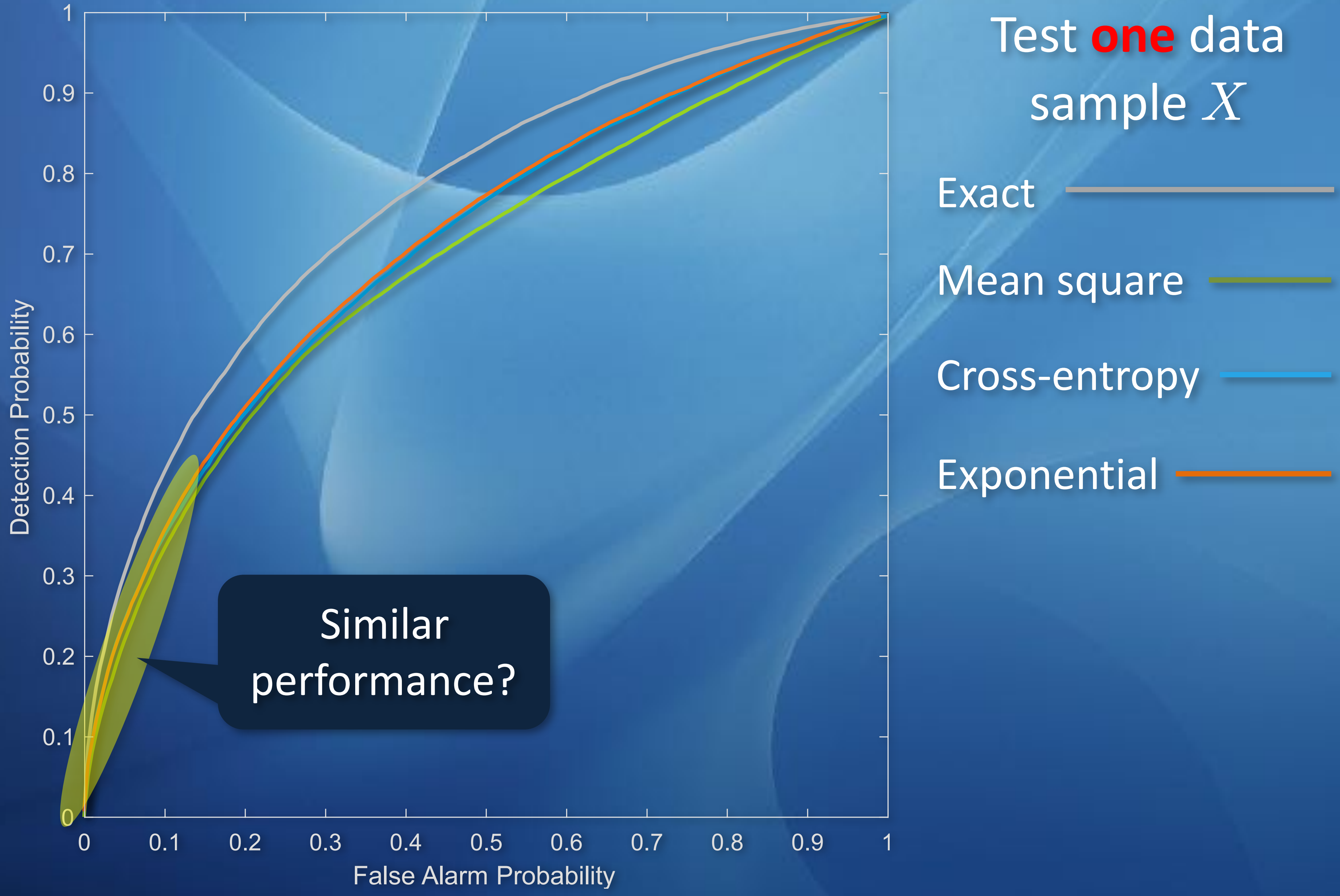
$$\mu_0 = 0, \Sigma_0 = \mathbf{I} \qquad \mu_1 = \frac{[1 \cdots 1]^\intercal}{\sqrt{10}}, \Sigma_1 = 1.2\mathbf{I}$$

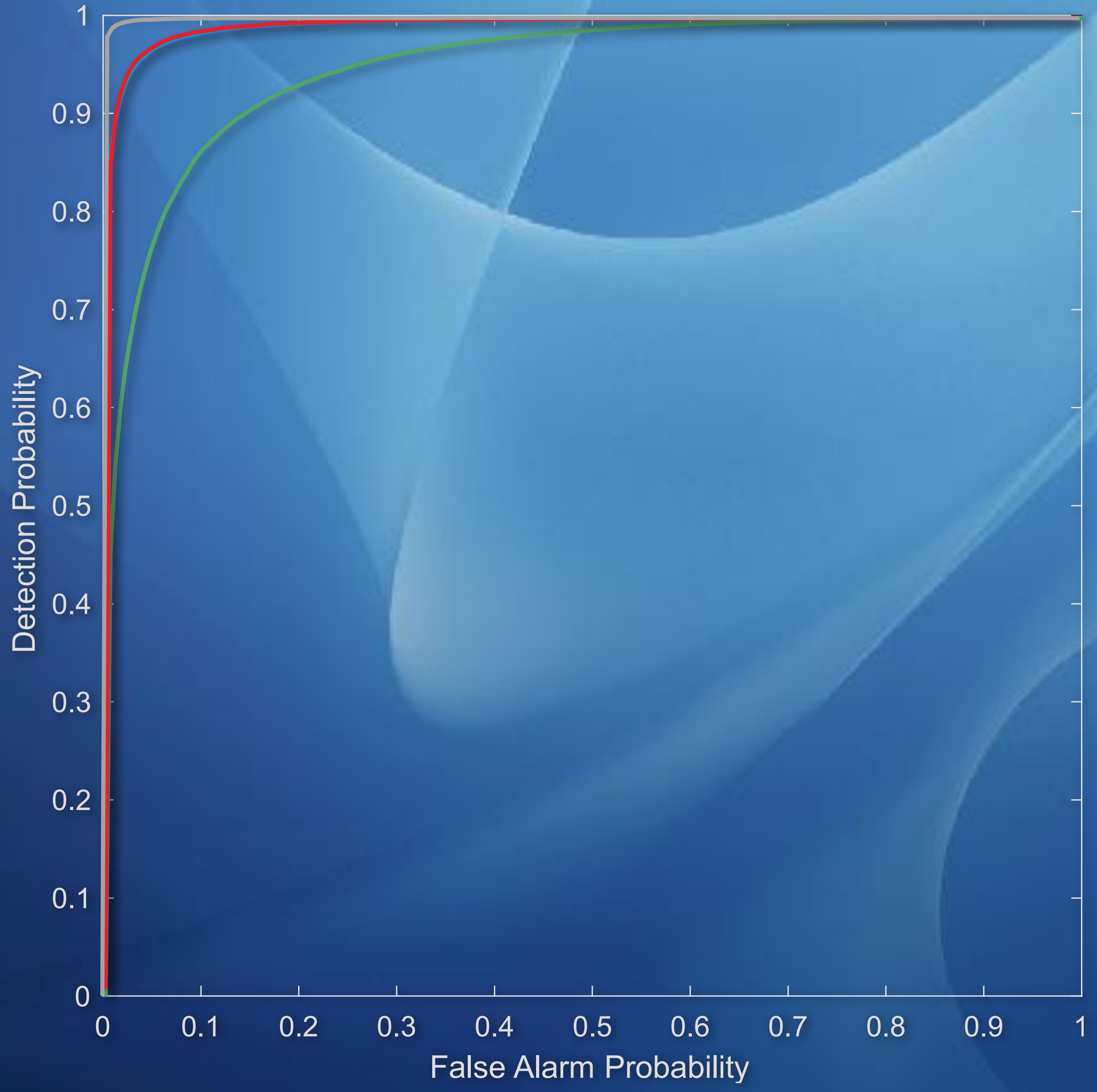
Input Hidden

Configuration : $10 \times 20 \times 1$ Output

$n_0 = n_1 = 100$ training data

Test **one** data sample X





Test **block of 20**

i.i.d. samples

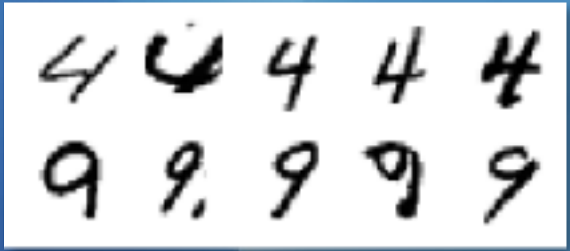
$X_1, \ldots, X_{20}$

Exact	▬
Mean square	▬
Cross-entropy	▬
Exponential	▬

Classification

MNIST database contains handwritten numbers

Isolate “4” and “9”. Handwritten versions resemble



28×28 gray scale

Transformed into

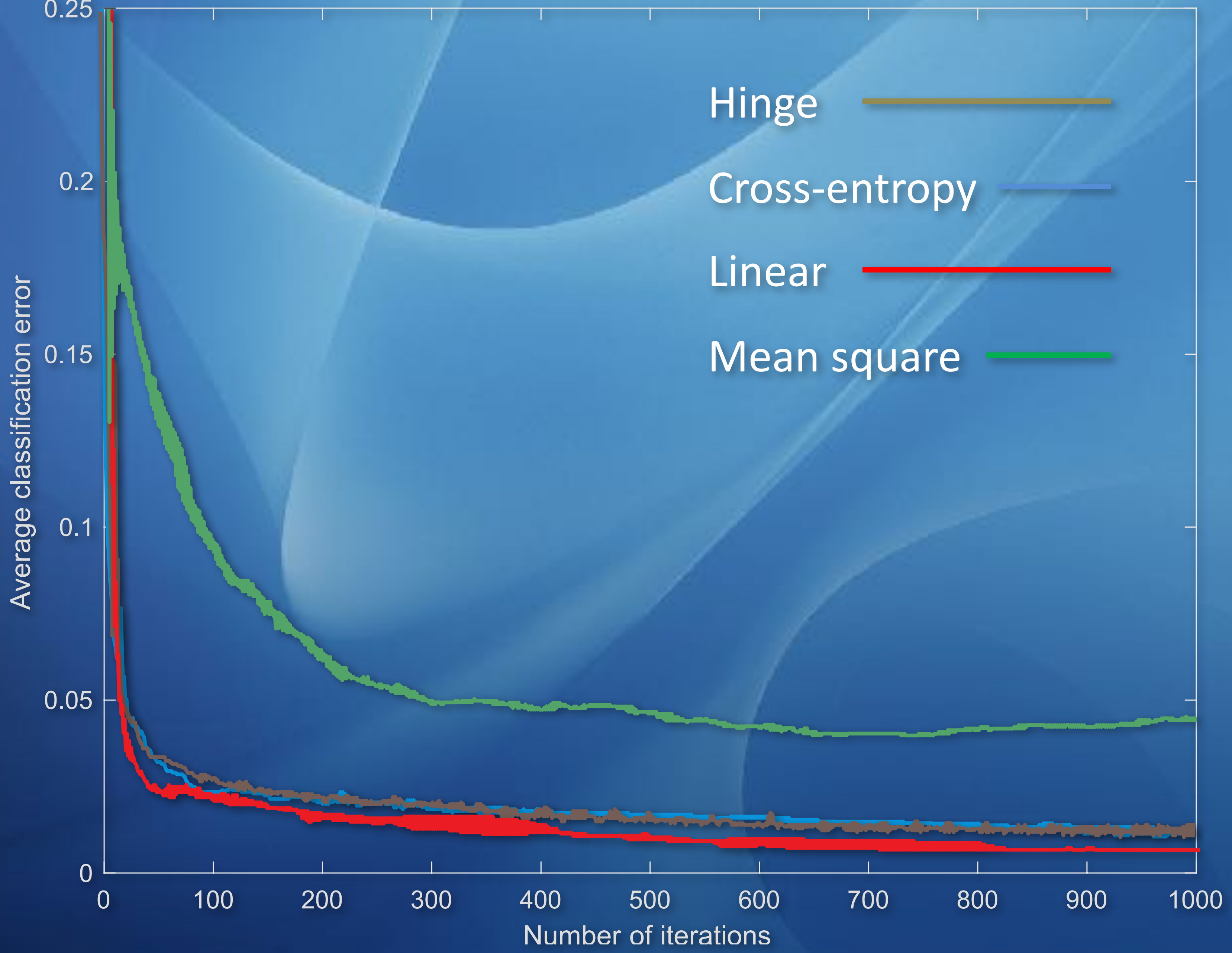
vector of length 728

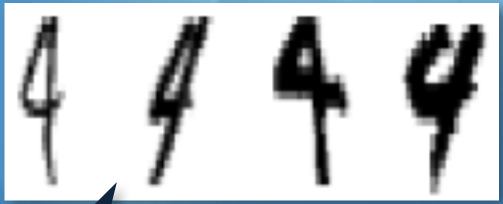
Build a **classifier** that distinguishes between the two

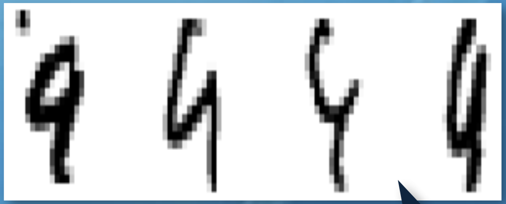
Training data: 5500 for “4” and 5500 for “9”

Neural network: $728 \times 300 \times 1$

Testing data: 982 for “4” and 1009 for “9”


Hinge
Cross-entropy
Linear
Mean square
Average classification error
0
0.05
0.1
0.15
0.2
0.25
0
100
200
300
400
500
600
700
800
900
1000
Number of iterations





“4” mistaken for “9”

“9” mistaken for “4”

Generalized likelihood ratio test

$\mathrm{H}_0: \quad X \sim \mathrm{f}_0(X)$

$\mathrm{H}_1: \quad X \sim \mathrm{f}_1(X, \vartheta)$

Testing Data (i.i.d.)

$\{X_1, X_2, \ldots, X_n\}$

$$\max_{\vartheta} \sum_{i=1}^{n} \log \frac{\mathrm{f}_1(X_i, \vartheta)}{\mathrm{f}_0(X_i)} \underset{<}{\overset{\geq}{=}} \nu$$

Data-Driven

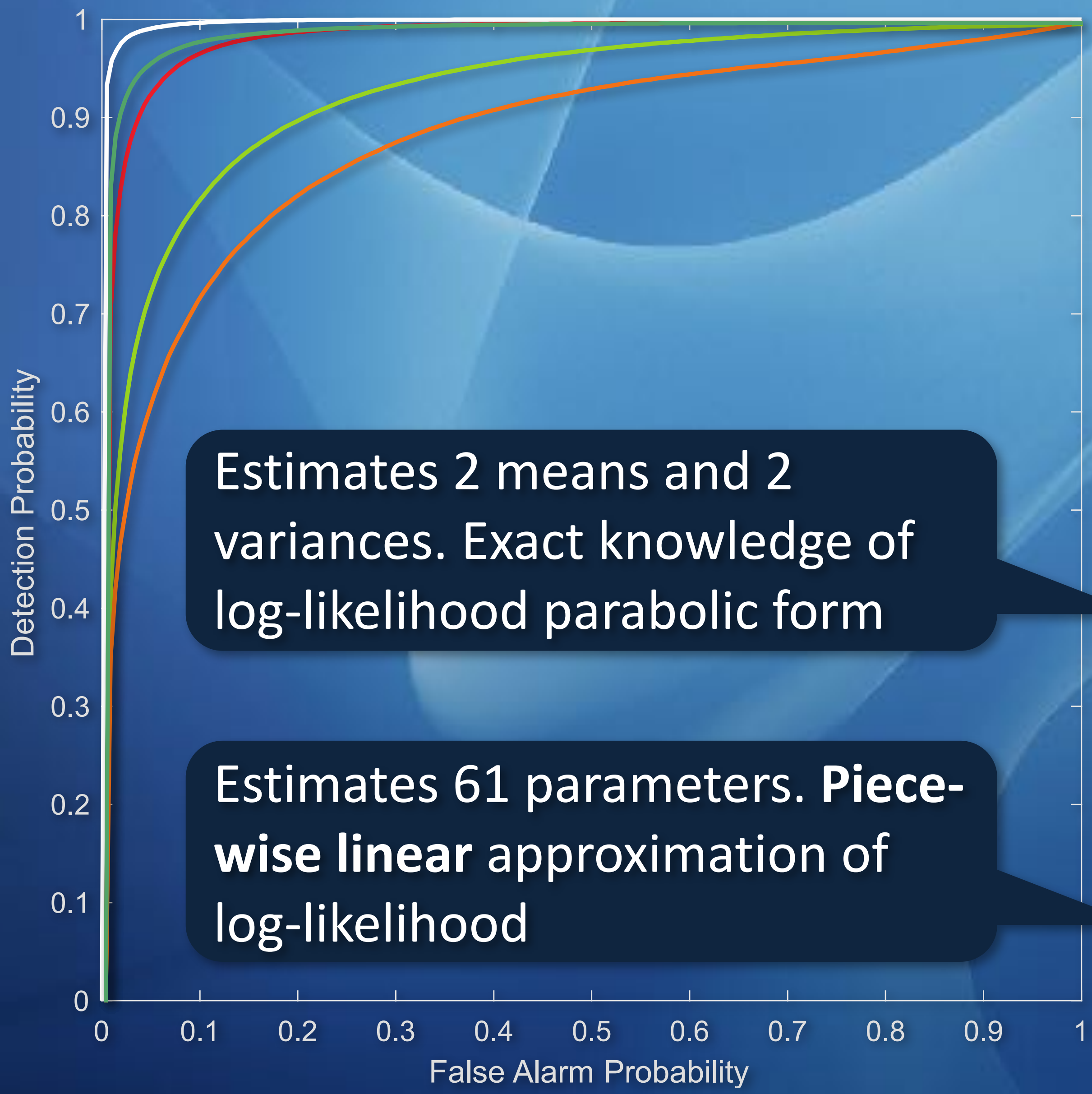
$\mathrm{H}_0: \quad \{X_1^0, X_2^0, \ldots, X_{n_0}^0\}$

$\mathrm{H}_1:$ No data

Testing Data (i.i.d.)

$\{X_1, X_2, \ldots, X_n\}$

Use the two sets to estimate log-likelihood ratio of a single X. Then form log-likelihood ratio of all testing data



Estimates 2 means and 2 variances. Exact knowledge of log-likelihood parabolic form

Estimates 61 parameters. **Piece-wise linear** approximation of log-likelihood

Data (i.i.d.)

$\{x_1, x_2, \ldots, x_n\}$

$f_0(x) \sim \mathcal{N}(0, 1)$

$f_1(x) \sim \mathcal{N}(0.4, 1.2)$

Exact $n=100$ ⬜

GLRT $n=100$ 🟩

GLRT $n=200$ 🟩

NN : $1 \times 20 \times 1$

NN $n=100$ 🟧

NN $n=200$ 🟥

Log-likelihood ratio for Markov processes

Can we approximate log-likelihood ratios of non-i.i.d.?

For example Markov processes?

$$\mathsf{f}(x_t,\ldots,x_1) =$$

$$\mathsf{f}(x_t|x_{t-1}\ldots,x_{t-k})\cdots\mathsf{f}(x_{k+1}|x_k\ldots,x_1)\mathsf{f}(x_k\ldots,x_1)$$

$$\mathsf{f}(x_t|x_{t-1}\ldots,x_{t-k}) = \frac{\mathsf{f}(x_t,x_{t-1}\ldots,x_{t-k})}{\mathsf{f}(x_{t-1}\ldots,x_{t-k})}$$

$$\log\frac{\mathsf{f}_1(x_t|x_{t-1}\ldots,x_{t-k})}{\mathsf{f}_0(x_t|x_{t-1}\ldots,x_{t-k})} =$$

$$\log\frac{\mathsf{f}_1(x_t,\ldots,x_{t-k})}{\mathsf{f}_0(x_t,\ldots,x_{t-k})} - \log\frac{\mathsf{f}_1(x_{t-1}\ldots,x_{t-k})}{\mathsf{f}_0(x_{t-1}\ldots,x_{t-k})}$$

$\mathsf{u}_{k+1}(x_t,\ldots,x_{t\text{-}k},\theta_{k+1})$

$\mathsf{u}_k(x_{t\text{-}1},\ldots,x_{t\text{-}k},\theta_k)$

Cumulative Sum (CUSUM) test (Change-detection)

$\{x_t\}$ acquired sequentially

$$\mathsf{S}_t = \max\{\mathsf{S}_{t-1}, 0\} + \log\frac{\mathsf{f}_1(x_t|x_{t-1}\ldots,x_{t-k})}{\mathsf{f}_0(x_t|x_{t-1}\ldots,x_{t-k})}$$

$$\hat{\mathsf{S}}_t = \max\{\hat{\mathsf{S}}_{t-1}, 0\} + \mathsf{u}_{k+1}(x_t,\ldots,x_{t-k},\theta_{k+1}) - \mathsf{u}_k(x_{t-1},\ldots,x_{t-k},\theta_k)$$

$$\mathsf{T} = \inf\{t>0 : \mathsf{S}_t \geq \nu\} \qquad \hat{\mathsf{T}} = \inf\{t>0 : \hat{\mathsf{S}}_t \geq \nu\}$$

Interested in

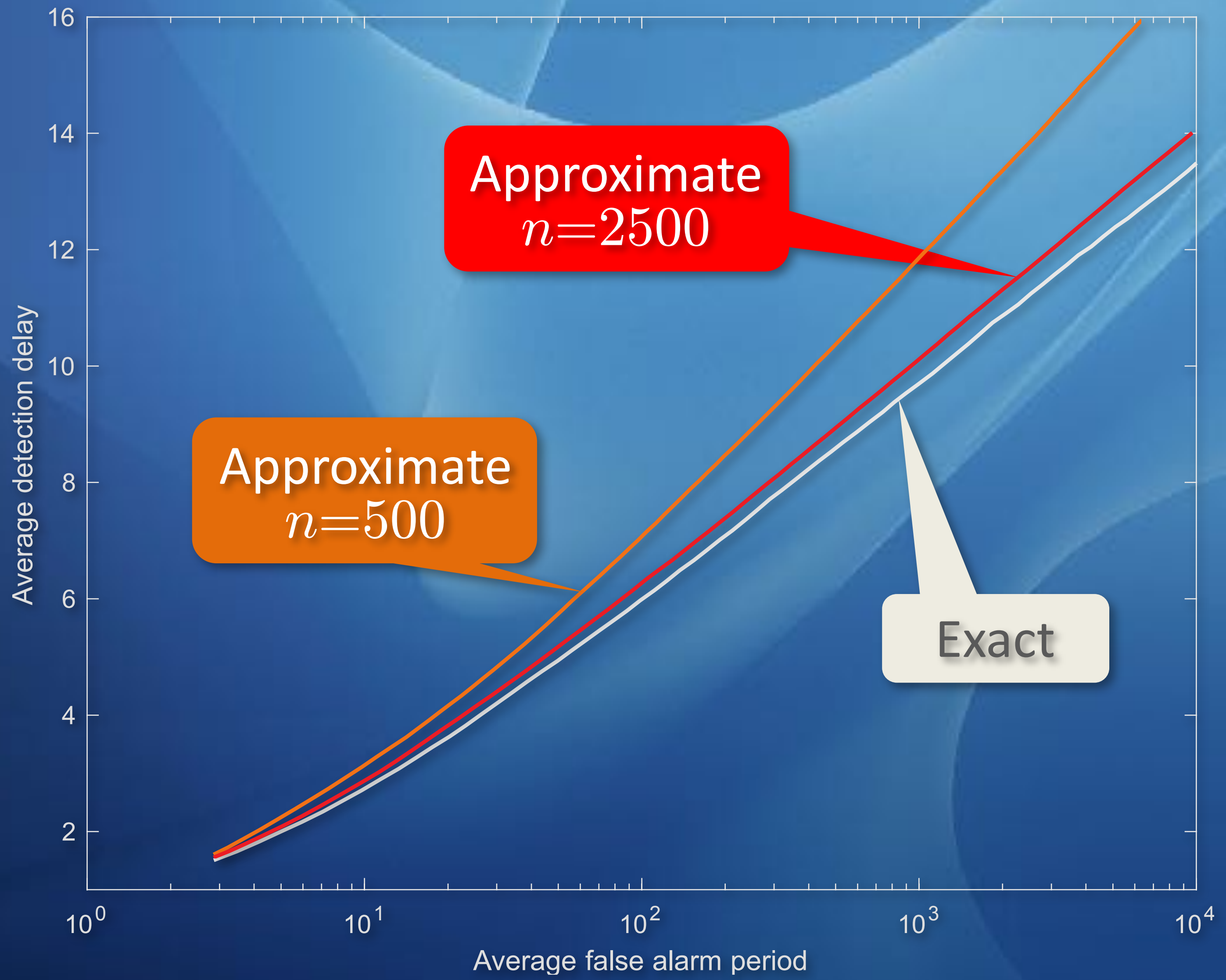
False Alarm Period (large)

$\mathsf{E}_0[\mathsf{T}]$ versus $\mathsf{E}_1[\mathsf{T}]$

$\mathsf{E}_0[\hat{\mathsf{T}}]$ versus $\mathsf{E}_1[\hat{\mathsf{T}}]$

Detection Delay (small)

$$\mathsf{f}_0(x_t|x_{t-1}) \sim \mathcal{N}(0,1) \quad \mathsf{f}_1(x_t|x_{t-1}) \sim \mathcal{N}\left(\mathsf{sgn}(x_{t-1})\sqrt{|x_{t-1}|}, 1\right)$$



$\mathsf{u}_2(x_t, x_{t-1}, \theta_2)$

$2 \times 50 \times 1$

$\mathsf{u}_1(x_t, \theta_1)$

$1 \times 20 \times 1$