CHAPTER

39 CHANGE DETECTION

CHAPTER CONTENTS

The objective of *change detection* is to investigate whether change exists between two data sets $\{\boldsymbol{x}_i\}_{i=1}^{n}$ and $\{\boldsymbol{x}'_{i'}\}_{i'=1}^{n'}$. In this chapter, two statistical approaches to change detection, *distributional change detection* and *structural change detection*, are explored. Below, $\{\boldsymbol{x}_i\}_{i=1}^{n}$ and $\{\boldsymbol{x}'_{i'}\}_{i'=1}^{n'}$ are assumed to be drawn independently from the probability distributions with densities $p(\boldsymbol{x})$ and $p'(\boldsymbol{x})$, respectively.

39.1 DISTRIBUTIONAL CHANGE DETECTION

Distributional change detection is aimed at identifying change in probability distributions behind $\{\boldsymbol{x}_i\}_{i=1}^{n}$ and $\{\boldsymbol{x}'_{i'}\}_{i'=1}^{n'}$. This can be achieved by estimating a *distance* or a *divergence* between $p(\boldsymbol{x})$ and $p'(\boldsymbol{x})$. As explained in Section 14.2, a distance satisfies four conditions: *non-negativity*, *symmetry*, *identity*, and the *triangle inequality*. On the other hand, a divergence is a pseudodistance that still acts like a distance, but it may violate some of the above conditions. In this section, divergence and distance measures between probability distributions that are useful for change detection and their approximators from samples $\{\boldsymbol{x}_i\}_{i=1}^{n}$ and $\{\boldsymbol{x}'_{i'}\}_{i'=1}^{n'}$ are introduced.

An Introduction to Statistical Machine Learning. DOI: 10.1016/B978-0-12-802121-7.00050-9

39.1.1 KL DIVERGENCE

The most popular divergence measure in statistics and machine learning would be the KL divergence (see Section 14.2), because of its compatible with MLE (see Chapter 12):

$$\mathrm{KL}(p\|p') = \int p(\boldsymbol{x}) \log \frac{p(\boldsymbol{x})}{p'(\boldsymbol{x})} \mathrm{d}\boldsymbol{x}.$$

Due to the log function which is sharp near zero, the KL divergence may allow sensitive detection of small change in probability distributions. Another advantage of the KL divergence is that it is *invariant* under input metric change, i.e., the value of the KL divergence does not change even if $\boldsymbol{x}$ is transformed to $\widetilde{\boldsymbol{x}}$ by *any* mapping [5].

The KL divergence can be approximated from samples $\{\boldsymbol{x}_i\}_{i=1}^n$ and $\{\boldsymbol{x}'_{i'}\}_{i'=1}^{n'}$ by the *direct density ratio estimator* introduced in Section 38.3. More specifically, an estimator $\widehat{w}(\boldsymbol{x})$ of the density ratio function,

$$w(\boldsymbol{x}) = \frac{p(\boldsymbol{x})}{p'(\boldsymbol{x})},$$

can be obtained by *KL density ratio estimation*. Then, the KL divergence can be immediately approximated as

$$\widehat{\mathrm{KL}} = \frac{1}{n} \sum_{i=1}^{n} \log \widehat{w}(\boldsymbol{x}_i).$$

However, due to the log function, estimation of the KL divergence is prone to be sensitive to *outliers*.

39.1.2 PEARSON DIVERGENCE

Ali-Silvey-Csiszár divergences [2, 35], which is also known as *f-divergences*, are generalization of the KL divergence defined as

$$F(p\|p') = \int p'(\boldsymbol{x}) f\left(\frac{p(\boldsymbol{x})}{p'(\boldsymbol{x})}\right) \mathrm{d}\boldsymbol{x},$$

where $f(t)$ is a *convex* function (see Fig. 8.3) such that $f(1) = 0$. It can be easily confirmed that, with $f(t) = t \log t$, the Ali-Silvey-Csiszár divergence is reduced to the KL divergence. Note that all Ali-Silvey-Csiszár divergences are invariant under input metric change.

The *Pearson divergence* [79] is a squared-loss variant of the KL divergence defined as the Ali-Silvey-Csiszár divergence with the squared function $f(t) = (t-1)^2$:

$$\mathrm{PE}(p\|p') = \int p'(\boldsymbol{x}) \left(\frac{p(\boldsymbol{x})}{p'(\boldsymbol{x})} - 1\right)^2 \mathrm{d}\boldsymbol{x}.$$

Since the Pearson divergence does not include the log function, it would be more robust against outliers. However, it still includes the density ratio function $p(\boldsymbol{x})/p'(\boldsymbol{x})$, which tends to be a sharp function and is possibly unbounded. Therefore, its accurate approximation is not straightforward in practice. As discussed in Section 33.2.2, this problem can be mitigated by considering the *relative density ratio* for $\beta \in [0,1]$:

$$w_\beta(\boldsymbol{x}) = \frac{p(\boldsymbol{x})}{\beta p(\boldsymbol{x}) + (1-\beta)p'(\boldsymbol{x})}.$$

The Pearson divergence extended using the relative density ratio is called the *relative Pearson divergence* [121]:

$$\begin{aligned}\mathrm{rPE}(p\|p') &= \mathrm{PE}(p\|\beta p + (1-\beta)p') \\ &= \int \Big(\beta p(\boldsymbol{x}) + (1-\beta)p'(\boldsymbol{x})\Big)\Big(w_\beta(\boldsymbol{x}) - 1\Big)^2 \mathrm{d}\boldsymbol{x}.\end{aligned}$$

The relative Pearson divergence can be approximated from samples $\{\boldsymbol{x}_i\}_{i=1}^n$ and $\{\boldsymbol{x}'_{i'}\}_{i'=1}^{n'}$ by the *direct relative density ratio estimator* introduced in Section 33.2.4. More specifically, an estimator $\widehat{w}_\beta(\boldsymbol{x})$ of the relative density ratio function $w_\beta(\boldsymbol{x})$ can be obtained by *LS relative density ratio estimation*. Then, the relative Pearson divergence can be approximated as

$$\widehat{\mathrm{rPE}} = \frac{1}{n}\sum_{i=1}^n \widehat{w}_\beta(\boldsymbol{x}_i) - 1,$$

which comes from the following expression of the relative Pearson divergence:

$$\mathrm{rPE}(p\|p') = \int p(\boldsymbol{x}) w_\beta(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} - 1.$$

The tuning parameter $\beta \in [0,1]$ controls the trade-off between sensitivity and robustness of the divergence measure, which should be appropriately chosen in practice.

39.1.3 L_2-DISTANCE

The L_2-*distance* is another standard distance measure between probability distributions:

$$L_2(p,p') = \int f(\boldsymbol{x})^2 \mathrm{d}\boldsymbol{x},$$

where

$$f(\boldsymbol{x}) = p(\boldsymbol{x}) - p'(\boldsymbol{x}).$$

The L_2-distance is a proper distance measure, and thus it is symmetric and satisfies the triangle inequality unlike the KL divergence and the (relative) Pearson divergence. Furthermore, the density difference $f(\boldsymbol{x})$ is always bounded as long as each

density is bounded. Therefore, the L_2-distance is stable, without the need of tuning any control parameter such as β in the relative Pearson divergence.

The L_2-distance can be approximated from samples $\{\boldsymbol{x}_i\}_{i=1}^{n}$ and $\{\boldsymbol{x}'_{i'}\}_{i'=1}^{n'}$ by *LS density difference estimation* [103] that directly estimates the density difference function $f(\boldsymbol{x})$ without estimating $p(\boldsymbol{x})$ and $p'(\boldsymbol{x})$ (see Section 37.4 for direct estimation of $p(\boldsymbol{x}, y) - p(\boldsymbol{x})p(y)$). More specifically, let us consider the following Gaussian density difference model:

$$f_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \sum_{j=1}^{n+n'} \alpha_j \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{c}_j\|^2}{2h^2}\right),$$

where $h > 0$ denotes the Gaussian bandwidth and

$$(\boldsymbol{c}_1, \ldots, \boldsymbol{c}_n, \boldsymbol{c}_{n+1}, \ldots, \boldsymbol{c}_{n+n'}) = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n, \boldsymbol{x}'_1, \ldots, \boldsymbol{x}'_{n'})$$

are the Gaussian centers. The parameters $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{n+n'})^\top$ are estimated by LS fitting to the true density difference function $f(\boldsymbol{x})$:

$$\min_{\boldsymbol{\alpha}} \int \left(f_{\boldsymbol{\alpha}}(\boldsymbol{x}) - f(\boldsymbol{x})\right)^2 \mathrm{d}\boldsymbol{x}.$$

Its empirical criterion where an irrelevant constant is ignored and the expectation is approximated by the sample average is given by

$$\min_{\boldsymbol{\alpha}} \left[\boldsymbol{\alpha}^\top \boldsymbol{U} \boldsymbol{\alpha} - 2\boldsymbol{\alpha}^\top \widehat{\boldsymbol{v}} + \lambda \|\boldsymbol{\alpha}\|^2\right],$$

where the ℓ_2-regularizer $\lambda\|\boldsymbol{\alpha}\|^2$ is included. $\boldsymbol{U}$ is the $(n + n') \times (n + n')$ matrix with the (j, j')th element defined by

$$\begin{aligned} U_{j,j'} &= \int \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{c}_j\|^2}{2h^2}\right) \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{c}_{j'}\|^2}{2h^2}\right) \mathrm{d}\boldsymbol{x} \\ &= (\pi h^2)^{d/2} \exp\left(-\frac{\|\boldsymbol{c}_j - \boldsymbol{c}_{j'}\|^2}{4h^2}\right), \end{aligned}$$

where d denotes the dimensionality of $\boldsymbol{x}$. $\widehat{\boldsymbol{v}}$ is the $(n + n')$-dimensional vector with the jth element defined by

$$\widehat{v}_j = \frac{1}{n} \sum_{i=1}^{n} \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{c}_j\|^2}{2h^2}\right) - \frac{1}{n'} \sum_{i'=1}^{n'} \exp\left(-\frac{\|\boldsymbol{x}'_{i'} - \boldsymbol{c}_j\|^2}{2h^2}\right).$$

This is a convex optimization problem, and the global optimal solution $\widehat{\boldsymbol{\alpha}}$ can be obtained *analytically* as

$$\widehat{\boldsymbol{\alpha}} = (\boldsymbol{U} + \lambda \boldsymbol{I})^{-1} \widehat{\boldsymbol{v}}.$$

```
n=200; x=randn(n,1); y=randn(n,1)+1;
hhs=2*[0.5 1 3].^2; ls=10.^[-2 -1 0]; m=5;
x2=x.^2; xx=repmat(x2,1,n)+repmat(x2',n,1)-2*x*x';
y2=y.^2; yx=repmat(y2,1,n)+repmat(x2',n,1)-2*y*x';
u=mod(randperm(n),m)+1; v=mod(randperm(n),m)+1;

for hk=1:length(hhs)
  hh=hhs(hk); k=exp(-xx/hh); r=exp(-yx/hh);
  U=(pi*hh/2)^(1/2)*exp(-xx/(2*hh));
  for i=1:m
    vh=mean(k(u~=i,:))'-mean(r(v~=i,:))';
    z=mean(k(u==i,:))-mean(r(v==i,:));
    for lk=1:length(ls)
      l=ls(lk); a=(U+l*eye(n))\vh; g(hk,lk,i)=a'*U*a-2*z*a;
end, end, end
[gl,ggl]=min(mean(g,3),[],2); [ghl,gghl]=min(gl);
L=ls(ggl(gghl)); HH=hhs(gghl);
k=exp(-xx/HH); r=exp(-yx/HH); vh=mean(k)'-mean(r)';
U=(pi*HH/2)^(1/2)*exp(-xx/(2*HH));
a=(U+L*eye(n))\vh; s=[k;r]*a; L2=a'*vh;
figure(1); clf; hold on; plot([x;y],s,'rx');
```

FIGURE 39.1

MATLAB code for LS density difference estimation.

The Gaussian width h and the regularization parameter λ may be optimized by cross validation with respect to the squared error criterion.

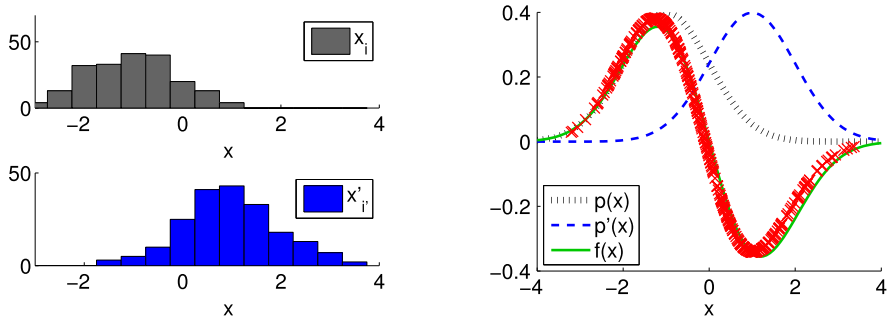
Finally, the L_2-distance can be approximated by

$$\widehat{L}^2 = \frac{1}{n}\sum_{i=1}^{n} f_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}_i) - \frac{1}{n'}\sum_{i'=1}^{n'} f_{\widehat{\boldsymbol{\alpha}}}(\boldsymbol{x}'_{i'}) = \widehat{\boldsymbol{v}}^\top \widehat{\boldsymbol{\alpha}}, \tag{39.1}$$

which comes from the following expression of the L_2-distance:

$$L_2(p,p') = \int \Big(p(\boldsymbol{x}) - p'(\boldsymbol{x})\Big) f(\boldsymbol{x}) \mathrm{d}\boldsymbol{x}.$$

A MATLAB code for LS density difference estimation is provided in Fig. 39.1, and its behavior is illustrated in Fig. 39.2. This shows that the density difference function can be accurately estimated.



(a) Samples (b) Densities and density differences

FIGURE 39.2

Example of LS density difference estimation. ×'s in the right plot show estimated density difference values at $\{\boldsymbol{x}_i\}_{i=1}^{n}$ and $\{\boldsymbol{x}'_{i'}\}_{i'=1}^{n'}$.

39.1.4 L_1-DISTANCE

The L_2-distance can be generalized to the class of L_s-distances for $s > 0$:

$$L_t(p,p') = \int \left|p(\boldsymbol{x}) - p'(\boldsymbol{x})\right|^s \mathrm{d}\boldsymbol{x}.$$

Among the class of L_s-distances, the L_1-*distance* is also a member of the class of the Ali-Silvey-Csiszár divergences with the absolute function $f(t) = |t-1|$ (see Section 39.1.2):

$$L_1(p,p') = \int \left|p(\boldsymbol{x}) - p'(\boldsymbol{x})\right|\mathrm{d}\boldsymbol{x} = \int p'(\boldsymbol{x}) \left|\frac{p(\boldsymbol{x})}{p'(\boldsymbol{x})} - 1\right| \mathrm{d}\boldsymbol{x}.$$

This implies that the L_1-distance is also invariant under input metric change.

Approximation of the L_1-distance from samples $\{\boldsymbol{x}_i\}_{i=1}^{n}$ and $\{\boldsymbol{x}'_{i'}\}_{i'=1}^{n'}$ can actually be performed by the (weighted) *support vector machine* introduced in Chapter 27 [75]. More specifically, since $|t| = \mathrm{sign}(t)\,t$, the L_1-distance can be expressed as

$$L_1(p,p') = \int \mathrm{sign}\Big(p(\boldsymbol{x}) - p'(\boldsymbol{x})\Big)\Big(p(\boldsymbol{x}) - p'(\boldsymbol{x})\Big)\mathrm{d}\boldsymbol{x},$$

where $\mathrm{sign}(t)$ denotes the sign function:

$$\mathrm{sign}(t) = \begin{cases} 1 & (t > 0), \\ 0 & (t = 0), \\ -1 & (t < 0). \end{cases}$$

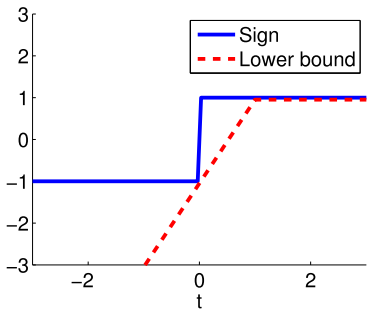


FIGURE 39.3

Lower bound of $\mathrm{sign}(t)$ by $-2\max(0,1-t)+1$.

For a density difference model $f_{\boldsymbol{\alpha}}(\boldsymbol{x})$ with parameter $\boldsymbol{\alpha}$, the L_1-distance $L_1(p,p')$ can be lower-bounded as

$$\begin{aligned} L_1(p,p') &\geq \max_{\boldsymbol{\alpha}} \int \mathrm{sign}\Big(f_{\boldsymbol{\alpha}}(\boldsymbol{x})\Big)\Big(p(\boldsymbol{x})-p'(\boldsymbol{x})\Big)\mathrm{d}\boldsymbol{x} \\ &= \max_{\boldsymbol{\alpha}} \left[\int \mathrm{sign}\Big(f_{\boldsymbol{\alpha}}(\boldsymbol{x})\Big)p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} + \int \mathrm{sign}\Big(-f_{\boldsymbol{\alpha}}(\boldsymbol{x})\Big)p'(\boldsymbol{x})\mathrm{d}\boldsymbol{x}\right], \end{aligned}$$

where the last term is due to $\mathrm{sign}(t) = -\mathrm{sign}(-t)$. As plotted in Fig. 39.3, the sign function can be lower-bounded by

$$-2\max(0,1-t)+1 = \begin{cases} 1 & (t>1), \\ 2t-1 & (t\leq 1). \end{cases}$$

Based on this, the L_1-distance $L_1(p,p')$ can be further lower-bounded as

$$\begin{aligned} L_1(p,p') \geq 2 - 2\min_{\boldsymbol{\alpha}} \Bigg[&\int \max\Big(0,1-f_{\boldsymbol{\alpha}}(\boldsymbol{x})\Big)p(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \\ &+ \int \max\Big(0,1+f_{\boldsymbol{\alpha}}(\boldsymbol{x})\Big)p'(\boldsymbol{x})\mathrm{d}\boldsymbol{x}\Bigg]. \end{aligned}$$

Let us employ a *linear-in-parameter* density difference model:

$$f_{\boldsymbol{\alpha}}(\boldsymbol{x}) = \sum_{j=1}^{b} \alpha_j \psi_j(\boldsymbol{x}) = \boldsymbol{\alpha}^\top \boldsymbol{\psi}(\boldsymbol{x}).$$

Then the empirical version of the above maximization problem (without irrelevant multiplicative and additive constants) is given by

$$\min_{\boldsymbol{\alpha}} \left[\frac{1}{n}\sum_{i=1}^{n} \max\Big(0,1-\boldsymbol{\alpha}^\top\boldsymbol{\psi}(\boldsymbol{x}_i)\Big) + \frac{1}{n'}\sum_{i'=1}^{n'} \max\Big(0,1+\boldsymbol{\alpha}^\top\boldsymbol{\psi}(\boldsymbol{x}'_{i'})\Big)\right].$$

Let us assign *class labels* $y_i = +1$ to $\boldsymbol{x}_i$ for $i = 1,\ldots,n$ and $y'_{i'} = -1$ to $\boldsymbol{x}'_{i'}$ for $i' = 1,\ldots,n'$. If $n = n'$, the above optimization problem agrees with *hinge loss minimization* (see Section 27.6):

$$\min_{\boldsymbol{\alpha}} \left[\sum_{i=1}^{n} \max\left(0, 1 - y_i \boldsymbol{\alpha}^\top \boldsymbol{\psi}(\boldsymbol{x}_i)\right) + \sum_{i'=1}^{n'} \max\left(0, 1 - y'_{i'} \boldsymbol{\alpha}^\top \boldsymbol{\psi}(\boldsymbol{x}'_{i'})\right) \right].$$

If $n \neq n'$, L_1-distance approximation corresponds to weighted hinge loss minimization with weight $1/n$ for $\{\boldsymbol{x}_i\}_{i=1}^n$ and $1/n'$ for $\{\boldsymbol{x}'_{i'}\}_{i'=1}^{n'}$.

The above formulation shows that the support vector machine is actually approximating the sign of the density difference. More specifically, let $p_+(\boldsymbol{x})$ and $p_-(\boldsymbol{x})$ be the probability density functions of samples in the positive class and negative class, respectively. Then, the support vector machine approximates

$$\mathrm{sign}\Big(p_+(\boldsymbol{x}) - p_-(\boldsymbol{x})\Big),$$

which is the optimal decision function. Thus, support vector classification can be interpreted as directly approximating the optimal decision function without estimating the densities $p_+(\boldsymbol{x})$ and $p_-(\boldsymbol{x})$.

39.1.5 MAXIMUM MEAN DISCREPANCY (MMD)

MMD [17] measures the distance between embeddings of probability distributions in a *reproducing kernel Hilbert space* [9].

More specifically, the MMD between p and p' is defined as

$$\begin{aligned}\mathrm{MMD}(p,p') = {} & \mathbb{E}_{\boldsymbol{x},\widetilde{\boldsymbol{x}}\sim p}[K(\boldsymbol{x},\widetilde{\boldsymbol{x}})] + \mathbb{E}_{\boldsymbol{x}',\widetilde{\boldsymbol{x}}'\sim p'}[K(\boldsymbol{x}',\widetilde{\boldsymbol{x}}')] \\ & - 2\mathbb{E}_{\boldsymbol{x}\sim p,\boldsymbol{x}'\sim p'}[K(\boldsymbol{x},\boldsymbol{x}')],\end{aligned}$$

where $K(\boldsymbol{x},\boldsymbol{x}')$ is a reproducing kernel, $\mathbb{E}_{\boldsymbol{x}\sim p}$ denotes the expectation with respect to $\boldsymbol{x}$ following density p. $\mathrm{MMD}(p,p')$ is always non-negative, and $\mathrm{MMD}(p,p') = 0$ if and only if $p = p'$ when $K(\boldsymbol{x},\boldsymbol{x}')$ is a *characteristic kernel* [45] such as the Gaussian kernel.

An advantage of MMD is that it can be directly approximated using samples as

$$\frac{1}{n^2}\sum_{i,\widetilde{i}=1}^{n} K(\boldsymbol{x}_i,\boldsymbol{x}_{\widetilde{i}}) + \frac{1}{n'^2}\sum_{i',\widetilde{i}'=1}^{n'} K(\boldsymbol{x}'_{i'},\boldsymbol{x}'_{\widetilde{i}'}) - \frac{2}{nn'}\sum_{i=1}^{n}\sum_{i'=1}^{n'} K(\boldsymbol{x}_i,\boldsymbol{x}'_{i'}).$$

Thus, no estimation is involved when approximating MMD from samples. However, it is not clear how to choose kernel functions in practice. Using the Gaussian kernel with bandwidth set at the median distance between samples is a popular heuristic [49], but this does not always work well in practice [50].

39.1.6 ENERGY DISTANCE

Another useful distance measure is the *energy distance* [106] introduced in Section 33.3.2:

$$D_{\mathrm{E}}(p,p') = \int_{\mathbb{R}^d} \|\boldsymbol{\varphi}_p(\boldsymbol{t}) - \boldsymbol{\varphi}_{p'}(\boldsymbol{t})\|^2 \left(\frac{\pi^{\frac{d+1}{2}}}{\Gamma\left(\frac{d+1}{2}\right)} \|\boldsymbol{t}\|^{d+1} \right)^{-1} \mathrm{d}\boldsymbol{t},$$

where $\|\cdot\|$ denotes the Euclidean norm, $\boldsymbol{\varphi}_p$ denotes the *characteristic function* (see Section 2.4.3) of p, $\Gamma(\cdot)$ is the *gamma function* (see Section 4.3), and d denotes the dimensionality of $\boldsymbol{x}$.

An important property of the energy distance is that it can be expressed as

$$D_{\mathrm{E}}(p,p') = 2\mathbb{E}_{\boldsymbol{x}\sim p,\boldsymbol{x}'\sim p'}\|\boldsymbol{x}-\boldsymbol{x}'\| - \mathbb{E}_{\boldsymbol{x},\widetilde{\boldsymbol{x}}\sim p}\|\boldsymbol{x}-\widetilde{\boldsymbol{x}}\| - \mathbb{E}_{\boldsymbol{x}',\widetilde{\boldsymbol{x}}'\sim p'}\|\boldsymbol{x}'-\widetilde{\boldsymbol{x}}'\|,$$

where $\mathbb{E}_{\boldsymbol{x}\sim p}$ denotes the expectation with respect to $\boldsymbol{x}$ following density p. This can be directly approximated using samples as

$$\frac{2}{nn'}\sum_{i=1}^{n}\sum_{i'=1}^{n'}\|\boldsymbol{x}_i - \boldsymbol{x}'_{i'}\| - \frac{1}{n^2}\sum_{i,\widetilde{i}=1}^{n}\|\boldsymbol{x}_i - \boldsymbol{x}_{\widetilde{i}}\| - \frac{1}{n'^2}\sum_{i',\widetilde{i}'=1}^{n'}\|\boldsymbol{x}'_{i'} - \boldsymbol{x}'_{\widetilde{i}'}\|.$$

Thus, no estimation and no tuning parameters are involved when approximating the energy distance from samples, which is a useful properties in practice.

Actually, it was shown [91] that the energy distance is a special case of MMD. Indeed, MMD with kernel function defined as

$$K(\boldsymbol{x},\boldsymbol{x}') = -\|\boldsymbol{x}-\boldsymbol{x}'\| + \|\boldsymbol{x}\| + \|\boldsymbol{x}'\|$$

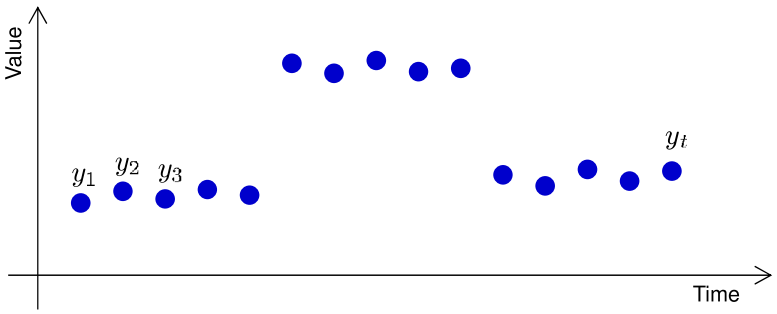
agrees with the energy distance.

39.1.7 APPLICATION TO CHANGE DETECTION IN TIME SERIES

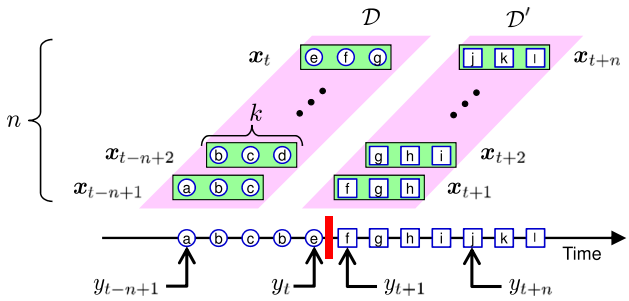
Let us consider the problem of *change detection in time series* (Fig. 39.4(a)). More specifically, given time series samples $\{y_i\}_{i=1}^N$, the objective is to identify whether change in probability distributions exists between y_t and y_{t+1} for some t. This problem can be tackled by estimating a distance (or a divergence) between the probability distributions of $\{y_i\}_{i=t-n+1}^{t}$ and $\{y_i\}_{i=t+1}^{t+n}$.

A challenge in change detection in time series is that samples $\{y_i\}_{i=1}^N$ are often dependent over time, which violates the presumption in this chapter. A practical approach to mitigate this problem is to *vectorize* data [60], as illustrated in Fig. 39.4(b). That is, instead of handling time series sample y_i as it is, its vectorization with k consecutive samples $\boldsymbol{x}_i = (y_i,\ldots,y_{i+k-1})^\top$ is considered, and a distance (or a divergence) is estimated from $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=t-n+1}^{t}$ and $\mathcal{D}' = \{\boldsymbol{x}_i\}_{i=t+1}^{t+n}$.

A MATLAB code for change detection in time series based on the energy distance is provided in Fig. 39.5, and its behavior is illustrated in Fig. 39.6. This shows that the energy distance well captures the distributional change in time series.



(a) Time series samples



(b) Vectorization to reduce time dependency

FIGURE 39.4

Change detection in time series.

39.2 STRUCTURAL CHANGE DETECTION

Distributional change detection introduced in the previous section focused on investigating whether change exists in probability distributions. The aim of *structural change detection* introduced in this section is to analyze change in the *dependency structure* between elements of d-dimensional variable $\boldsymbol{x} = (x^{(1)}, \ldots, x^{(d)})^\top$.

39.2.1 SPARSE MLE

Let us consider a *Gaussian Markov network*, which is a d-dimensional Gaussian model with expectation zero (Section 6.2):

$$q(\boldsymbol{x}; \boldsymbol{\Theta}) = \frac{\det(\boldsymbol{\Theta})^{1/2}}{(2\pi)^{d/2}} \exp\left(-\frac{1}{2}\boldsymbol{x}^\top \boldsymbol{\Theta} \boldsymbol{x}\right),$$

where not the variance-covariance matrix, but its inverse called the *precision matrix* is parameterized by $\boldsymbol{\Theta}$. If $\boldsymbol{\Theta}$ is regarded as an *adjacency matrix*, the Gaussian Markov network can be visualized as a *graph* (see Fig. 39.7). An advantage of this precision-based parameterization is that the connectivity governs conditional independence.

```
N=300; k=5; n=10; m=N-k+1; E=nan(1,N);
y=zeros(1,N); y(101:200)=3; y=y+randn(1,N);
%y=sin([1:N]/2); y(101:200)=sin([101:200]);
x=toeplitz(y); x=x(1:k,1:m); x2=sum(x.^2);
D=sqrt(repmat(x2',1,m)+repmat(x2,m,1)-2*x'*x);
for t=n:N-n-k+1
  a=[t-n+1:t]; b=[t+1:t+n];
  E(t)=2*mean(mean(D(a,b)))-mean(mean(D(a,a))) ...
       -mean(mean(D(b,b)));
end

figure(1); clf; hold on; plot(y,'b-'); plot(E,'r--');
legend('Time series','Energy distance')
```

FIGURE 39.5

MATLAB code for change detection in time series based on the energy distance.

For example, in the Gaussian Markov network illustrated in the left-hand side of Fig. 39.7, $x^{(1)}$ and $x^{(2)}$ are connected via $x^{(3)}$. This means that $x^{(1)}$ and $x^{(2)}$ are conditionally independent given $x^{(3)}$.

Suppose that $\{\boldsymbol{x}_i\}_{i=1}^{n}$ and $\{\boldsymbol{x}'_{i'}\}_{i'=1}^{n'}$ are drawn independently from the Gaussian Markov networks with precision matrices $\boldsymbol{\Theta}$ and $\boldsymbol{\Theta}'$, respectively. Then analyzing $\boldsymbol{\Theta} - \boldsymbol{\Theta}'$ allows us to identify the change in Markov network structure (see Fig. 39.7 again).

A sparse estimate of $\boldsymbol{\Theta}$ may be obtained by MLE with the ℓ_1-*constraint* (see Chapter 24):

$$\max_{\boldsymbol{\Theta}} \sum_{i=1}^{n} \log q(\boldsymbol{x}_i; \boldsymbol{\Theta}) \;\text{ subject to } \|\boldsymbol{\Theta}\|_1 \leq R^2,$$

where $R \geq 0$ is the radius of the ℓ_1-ball. This method is also referred to as the *graphical lasso* [44].

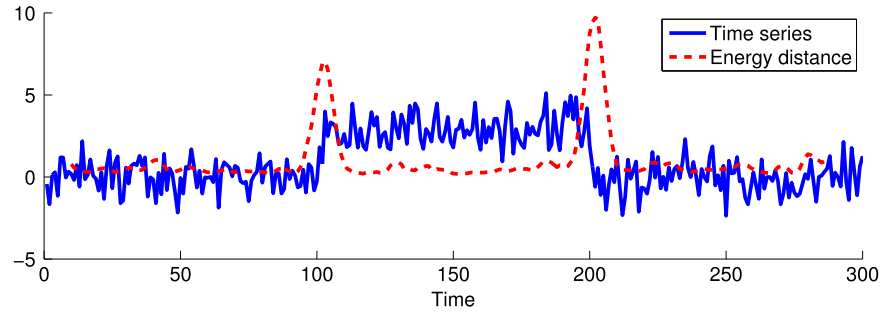
The derivative of $\log q(\boldsymbol{x}; \boldsymbol{\Theta})$ with respect to $\boldsymbol{\Theta}$ is given by

$$\frac{\partial \log q(\boldsymbol{x}; \boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} = \frac{1}{2}\boldsymbol{\Theta}^{-1} - \frac{1}{2}\boldsymbol{x}\boldsymbol{x}^{\top},$$

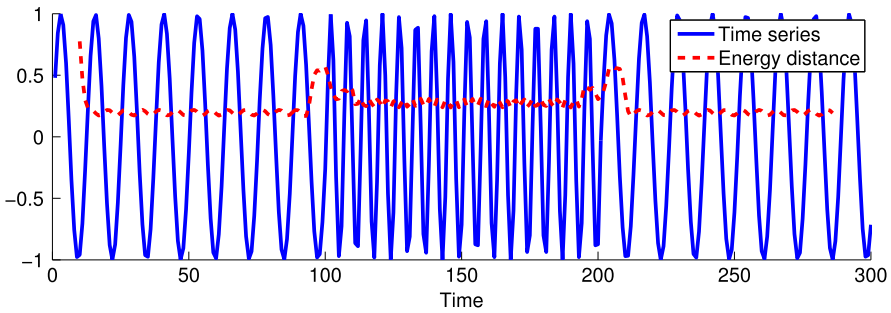
where the following formulas are used for its derivation:

$$\frac{\partial \log \det(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} = \boldsymbol{\Theta}^{-1} \quad \text{and} \quad \frac{\partial \boldsymbol{x}^{\top}\boldsymbol{\Theta}\boldsymbol{x}}{\partial \boldsymbol{\Theta}} = \boldsymbol{x}\boldsymbol{x}^{\top}.$$

A MATLAB code of a gradient-projection algorithm of ℓ_1-constraint MLE for Gaussian Markov networks is given in Fig. 39.8, where projection onto the ℓ_1-ball is computed by the method developed in [39].



(a) Bias change



(b) Frequency change

FIGURE 39.6

Examples of change detection in time series based on the energy distance.

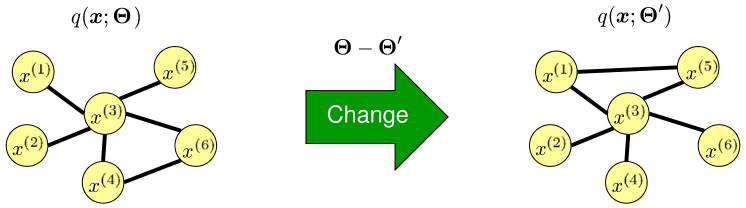


FIGURE 39.7

Structural change in Gaussian Markov networks.

For the true precision matrices

$$\boldsymbol{\Theta} = \begin{pmatrix} 2 & 0 & 1 \\ 0 & 2 & 0 \\ 1 & 0 & 2 \end{pmatrix} \quad \text{and} \quad \boldsymbol{\Theta}' = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 2 \end{pmatrix},$$

```
TT=[2 0 1; 0 2 0; 1 0 2];
%TT=[2 0 0; 0 2 0; 0 0 2];
%TT=[2 1 0; 1 2 1; 0 1 2];
%TT=[2 0 1; 0 2 1; 1 1 2];
d=3; n=50; x=TT^(-1/2)*randn(d,n); S=x*x'/n;
T0=eye(d); C=5; e=0.1;
for o=1:100000
  T=T0+e*(inv(T0)-S);
  T(:)=L1BallProjection(T(:),C);
  if norm(T-T0)<0.00000001, break, end
  T0=T;
end
T, TT
```

```
function w=L1BallProjection(x,C)

u=sort(abs(x),'descend'); s=cumsum(u);
r=find(u>(s-C)./(1:length(u))',1,'last');
w=sign(x).*max(0,abs(x)-max(0,(s(r)-C)/r));
```

FIGURE 39.8

MATLAB code of a gradient-projection algorithm of ℓ_1-constraint MLE for Gaussian Markov networks. The bottom function should be saved as "`L1BallProjection.m`."

sparse MLE gives

$$\widehat{\mathbf{\Theta}} = \begin{pmatrix} 1.382 & 0 & 0.201 \\ 0 & 1.788 & 0 \\ 0.201 & 0 & 1.428 \end{pmatrix} \quad \text{and} \quad \widehat{\mathbf{\Theta}}' = \begin{pmatrix} 1.617 & 0 & 0 \\ 0 & 1.711 & 0 \\ 0 & 0 & 1.672 \end{pmatrix}.$$

Thus, the true sparsity patterns of $\mathbf{\Theta}$ and $\mathbf{\Theta}'$ (in off-diagonal elements) can be successfully recovered. Since

$$\mathbf{\Theta} - \mathbf{\Theta}' = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \quad \text{and} \quad \widehat{\mathbf{\Theta}} - \widehat{\mathbf{\Theta}}' = \begin{pmatrix} -0.235 & 0 & 0.201 \\ 0 & 0.077 & 0 \\ 0.201 & 0 & -0.244 \end{pmatrix},$$

change in sparsity patterns (in off-diagonal elements) can be correctly identified.

On the other hand, when the true precision matrices are

$$\boldsymbol{\Theta} = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix} \quad \text{and} \quad \boldsymbol{\Theta}' = \begin{pmatrix} 2 & 0 & 1 \\ 0 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix},$$

sparse MLE gives

$$\widehat{\boldsymbol{\Theta}} = \begin{pmatrix} 1.303 & 0.348 & 0 \\ 0.348 & 1.157 & 0.240 \\ 0 & 0.240 & 1.365 \end{pmatrix} \quad \text{and} \quad \widehat{\boldsymbol{\Theta}}' = \begin{pmatrix} 1.343 & 0 & 0.297 \\ 0 & 1.435 & 0.236 \\ 0.297 & 0.236 & 1.156 \end{pmatrix}.$$

Thus, the true sparsity patterns of $\boldsymbol{\Theta}$ and $\boldsymbol{\Theta}'$ can still be successfully recovered. However, since

$$\boldsymbol{\Theta} - \boldsymbol{\Theta}' = \begin{pmatrix} 0 & 1 & -1 \\ 1 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix} \quad \text{and} \quad \widehat{\boldsymbol{\Theta}} - \widehat{\boldsymbol{\Theta}}' = \begin{pmatrix} -0.040 & 0.348 & -0.297 \\ 0.348 & -0.278 & 0.004 \\ -0.297 & 0.004 & 0.209 \end{pmatrix},$$

change in sparsity patterns was not correctly identified (although 0.004 is reasonably close to zero). This shows that, when a nonzero unchanged edge exists, say $\Theta_{k,k'} = \Theta'_{k,k'} > 0$ for some k and k', it is difficult to identify this unchanged edge because $\widehat{\Theta}_{k,k'} \approx \widehat{\Theta}'_{k,k'}$ does not necessarily hold by separate sparse MLE from $\{\boldsymbol{x}_i\}_{i=1}^n$ and $\{\boldsymbol{x}'_{i'}\}_{i'=1}^{n'}$.

39.2.2 SPARSE DENSITY RATIO ESTIMATION

As illustrated above, sparse MLE can perform poorly in structural change detection. Another limitation of sparse MLE is the Gaussian assumption. A Gaussian Markov network can be extended to a non-Gaussian model as

$$q(\boldsymbol{x};\boldsymbol{\theta}) = \frac{\overline{q}(\boldsymbol{x};\boldsymbol{\theta})}{\int \overline{q}(\boldsymbol{x};\boldsymbol{\theta})\mathrm{d}\boldsymbol{x}},$$

where, for a *feature vector* $\boldsymbol{f}(x, x')$,

$$\overline{q}(\boldsymbol{x};\boldsymbol{\theta}) = \exp\left(\sum_{k \geq k'} \boldsymbol{\theta}_{k,k'}^\top \boldsymbol{f}(x^{(k)}, x^{(k')})\right).$$

This model is reduced to the Gaussian Markov network if

$$\boldsymbol{f}(x, x') = -\frac{1}{2} x x',$$

while higher-order correlations can be captured by considering higher-order terms in the feature vector. However, applying sparse MLE to non-Gaussian Markov networks is not straightforward in practice because the normalization term $\int \overline{q}(\boldsymbol{x};\boldsymbol{\theta})\mathrm{d}\boldsymbol{x}$ is often computationally intractable.

To cope with these limitations, let us handle the change in parameters, $\boldsymbol{\theta}_{k,k'}-\boldsymbol{\theta}'_{k,k'}$, directly via the following density ratio function:

$$\frac{q(\boldsymbol{x};\boldsymbol{\theta})}{q(\boldsymbol{x};\boldsymbol{\theta}')} \propto \exp\left(\sum_{k\geq k'}(\boldsymbol{\theta}_{k,k'}-\boldsymbol{\theta}'_{k,k'})^\top \boldsymbol{f}(x^{(k)},x^{(k')})\right).$$

Based on this expression, let us consider the following density ratio model:

$$r(\boldsymbol{x};\boldsymbol{\alpha}) = \frac{\exp\left(\sum_{k\geq k'}\boldsymbol{\alpha}_{k,k'}^\top \boldsymbol{f}(x^{(k)},x^{(k')})\right)}{\int p'(\boldsymbol{x})\exp\left(\sum_{k\geq k'}\boldsymbol{\alpha}_{k,k'}^\top \boldsymbol{f}(x^{(k)},x^{(k')})\right)\mathrm{d}\boldsymbol{x}}, \tag{39.2}$$

where $\boldsymbol{\alpha}_{k,k'}$ is the difference of parameters:

$$\boldsymbol{\alpha}_{k,k'} = \boldsymbol{\theta}_{k,k'} - \boldsymbol{\theta}'_{k,k'}.$$

$p'(\boldsymbol{x})$ in the denominator of Eq. (39.2) comes from the fact that $r(\boldsymbol{x};\boldsymbol{\alpha})$ approximates $p(\boldsymbol{x})/p'(\boldsymbol{x})$ and thus the normalization constraint,

$$\int r(\boldsymbol{x};\boldsymbol{\alpha})p'(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = 1,$$

is imposed.

Let us learn the parameters $\{\boldsymbol{\alpha}_{k,k'}\}_{k\geq k'}$ by a *group-sparse* variant (see Section 24.4.4) of *KL density ratio estimation* explained in Section 38.3 [69]:

$$\begin{aligned}\min_{\{\boldsymbol{\alpha}_{k,k'}\}_{k\geq k'}} \quad & \log\frac{1}{n'}\sum_{i'=1}^{n'}\exp\left(\sum_{k\geq k'}\boldsymbol{\alpha}_{k,k'}^\top \boldsymbol{f}(x_{i'}^{\prime(k)},x_{i'}^{\prime(k')})\right)\\ & -\frac{1}{n}\sum_{i=1}^{n}\sum_{k\geq k'}\boldsymbol{\alpha}_{k,k'}^\top \boldsymbol{f}(x_i^{(k)},x_i^{(k')})\\ & \text{subject to}\ \sum_{k\geq k'}\|\boldsymbol{\alpha}_{k,k'}\| \leq R^2,\end{aligned}$$

where $R \geq 0$ controls the sparseness of the solution.

A MATLAB code of a gradient-projection algorithm of sparse KL density ratio estimation for Gaussian Markov networks is given in Fig. 39.9. For the true precision matrices

$$\boldsymbol{\Theta}-\boldsymbol{\Theta}' = \begin{pmatrix}2&0&1\\0&2&0\\1&0&2\end{pmatrix} - \begin{pmatrix}2&0&0\\0&2&0\\0&0&2\end{pmatrix} = \begin{pmatrix}0&0&1\\0&0&0\\1&0&0\end{pmatrix},$$

```
Tp=[2 0 1; 0 2 0; 1 0 2]; Tq=[2 0 0; 0 2 0; 0 0 2];
Tp=[2 1 0; 1 2 1; 0 1 2]; Tq=[2 0 1; 0 2 1; 1 1 2];
d=3; n=50; xp=Tp^(-1/2)*randn(d,n); Sp=xp*xp'/n;
xq=Tq^(-1/2)*randn(d,n); A0=eye(d); C=1; e=0.1;
for o=1:1000000
  U=exp(sum((A0*xq).*xq));
  A=A0-e*((repmat(U,[d 1]).*xq)*xq'/sum(U)-Sp);
  A(:)=L1BallProjection(A(:),C);
  if norm(A-A0)<0.00000001, break, end
  A0=A;
end
-2*A, Tp-Tq
```

FIGURE 39.9

MATLAB code of a gradient-projection algorithm of ℓ_1-constraint KL density ratio estimation for Gaussian Markov networks. "`L1BallProjection.m`" is given in Fig. 39.8.

sparse KL density ratio estimation gives

$$\begin{pmatrix} 0 & 0 & 1.000 \\ 0 & 0 & 0 \\ 1.000 & 0 & 0 \end{pmatrix}.$$

This implies that the change in sparsity patterns can be correctly identified.

Even when nonzero unchanged edges exist as

$$\boldsymbol{\Theta} - \boldsymbol{\Theta}' = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix} - \begin{pmatrix} 2 & 0 & 1 \\ 0 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix} = \begin{pmatrix} 0 & 1 & -1 \\ 1 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix},$$

sparse KL density ratio estimation gives

$$\begin{pmatrix} 0 & 0.707 & -0.293 \\ 0.707 & 0 & 0 \\ -0.293 & 0 & 0 \end{pmatrix}.$$

Thus, the change in Markov network structure can still be correctly identified.