Algorithm 10.1 Sampling procedure to construct the confidence interval for the binning method. Set $M_w = 1,000$, $M_l = 10,000$, $N_w = 100$, and $N_l = 100$.

1. Draw $M_w \times N_w$ samples from the joint distribution $f(\tilde{V}, \tilde{s}|\mathcal{D}_V, \mathcal{D}_s)$ of wind characteristics $(\tilde{V}, \tilde{s})$, where the tilde notation indicates a sampled quantity. Please cross reference Algorithm 10.4 for specific models and steps to draw samples for wind characteristics $(\tilde{V}, \tilde{s})$.

2. Using the data in a bin, employ an MLE method to estimate μ and σ in the GEV, while fixing ξ. Draw a sample of μ and σ for that specific bin from a multivariate normal distributions taking the MLE as its mean and the inverse of the negative of Hessian matrix as its covariance matrix. Not all the bins have data. For those which do not have data, its μ and σ are a weighted average of all non-empty bins with the weight related to the inverse squared distance between bins. Collectively, $\mathbf{\Phi}_c$ contains the μ's and σ's from all the bins.

3. Decide which bins the wind characteristic samples $(\tilde{V}, \tilde{s})$'s fall into. Based on the specific bin in which a sample of $(\tilde{V}, \tilde{s})$ falls, the corresponding μ and σ in $\mathbf{\Phi}_c$ is chosen. Doing so yields the short-term distribution $f(\tilde{z}|\tilde{V}, \tilde{s}, \mathbf{\Phi}_c)$ for that specific bin.

4. Draw N_l samples of $\tilde{z}$ from $f(\tilde{z}|\tilde{V}, \tilde{s}, \mathbf{\Phi}_c)$ for each of the total $M_w \times N_w$ samples of $(\tilde{V}, \tilde{s})$. This produces a total of $M_w \times N_w \times N_l$ samples of $\tilde{z}$.

5. One can then compute the quantile value $l_T[\mathbf{\Phi}_c]$ corresponding to P_T.

6. Repeat the above procedure M_l times to get the median and confidence interval of l_T.

load. In practice, how many bins to use is also under debate, and there is not yet a consensus. The answer to the action of binning sometimes depends on the amount of data—if an analyst has more data, he/she affords to use more bins; otherwise, fewer bins.

The popularity of the binning method in industrial practice is due to the simplicity of its idea and procedure. However, simplicity of a procedure should not be mistaken as simplicity of the resulting model. Suppose that one uses a 6×10 grid to bin the two-dimensional wind covariates and fixes the shape parameter ξ across the bins (a common practice in the industry). The binning method yields 60 local GEV distributions, each of which has two parameters, translating to a total of 121 parameters for the overall model (counting the fixed ξ as well). A model having 121 parameters is not a simple model. The combination of the rigidity of compartmentalization and the unintended high model complexity renders the binning method not scalable and less effective.

10.4 BAYESIAN SPLINE-BASED GEV MODEL

Lee et al. [131] present a Bayesian spline method for estimating the extreme load on wind turbines. The spline method is essentially a method supporting an inhomogeneous GEV distribution to capture the nonlinear relationship between the load response and the wind-related covariates. Such treatment avoids binning the data. The underlying spline model connect all the bins across the whole wind profile, so that load and wind data are pooled together to produce better estimates. The merit of the spline model is demonstrated in Section 10.6 by applying it to three sets of turbine load response data and making comparisons with the binning method.

10.4.1 Conditional Load Model

Recall that in the binning method, a homogeneous GEV distribution is used to model the short-term load distribution in a bin, for it appears reasonable to assume stationarity if the chosen weather bin is narrow enough. A finite number of the homogeneous GEV distributions are then stitched together to represent the nonstationary nature across the entire wind profile; see Fig. 10.5. What Lee et al. [131] propose is to abandon the bins and instead use an inhomogeneous GEV distribution whose parameters are not constant but depend on weather conditions.

Consider 10-minute maximum loads, $z_1, \ldots, z_n$, with corresponding covariate variables $\boldsymbol{x}_1 = (V_1, s_1)$, …, $\boldsymbol{x}_n = (V_n, s_n)$, as defined in Eq. 10.1. Let us consider modeling z_i conditional on $\boldsymbol{x}$, such that

$$z_i|\boldsymbol{x}_i \sim \text{GEV}(\mu(\boldsymbol{x}_i), \sigma(\boldsymbol{x}_i), \xi), \quad \sigma(\cdot) > 0, \tag{10.9}$$

where the location parameter μ and scale parameter σ in this GEV distribution are a nonlinear function of wind characteristics $\boldsymbol{x}$. The shape parameter ξ is fixed across the wind profile, while its value will still be estimated using

the data from a specific wind turbine. The reason behind the fixed ξ is to keep the final model from becoming overly flexible. Too much flexibility could cause complexity in model fitting and parameter estimation.

Let us denote $\mu(\boldsymbol{x}_i)$ and $\sigma(\boldsymbol{x}_i)$ by

$$\mu(\boldsymbol{x}_i) = q(\boldsymbol{x}_i), \tag{10.10}$$

$$\sigma(\boldsymbol{x}_i) = \exp(g(\boldsymbol{x}_i)), \tag{10.11}$$

where an exponential function is used in Eq. 10.11 to ensure the positivity of the scale parameter. To capture the nonlinearity between the load response and the wind-related covariates, Lee et al. [131] model $q(\cdot)$ and $g(\cdot)$ using a Bayesian MARS model [46, 47]. Recall the discussion about splines in Section 5.3.3. A shortcoming of the spline methods is its lack of scalability to model multivariate inputs. One of the methods that addresses this issue is the MARS model [68], which uses an additive model structure, allowing factor interactions to be added through a hierarchical inclusion of interaction terms for the purpose of accomplishing scalability. The Bayesian MARS model is basically a MARS model but includes the number and locations of knots as part of its model parameters and determines these from observed data.

Lee at al. [131] state that they explore simple approaches based on polynomial models for modeling $\mu(\boldsymbol{x})$ and $\sigma(\boldsymbol{x})$. It turns out that polynomial-based approaches lack the flexibility of adapting to the datasets from different types of turbines. Due to the nonlinearity around the rated wind speed and the limited amount of data under high wind speeds, polynomial-based approaches perform poorly in those regions that are generally important for capturing the maximum load. Spline models, on the other hand, appear to work better than a global polynomial model, because they have more supporting points spreading over the input region.

The Bayesian MARS models, i.e., $q(\boldsymbol{x})$ for the location parameter μ and $g(\boldsymbol{x})$ for the scale parameter σ, are represented as a linear combination of the basis functions $B_k^{\mu}(\boldsymbol{x})$ and $B_k^{\sigma}(\boldsymbol{x})$, respectively, such that

$$q(\boldsymbol{x}) = \sum_{k=1}^{K_\mu} \beta_k B_k^{\mu}(\boldsymbol{x}), \quad \text{and} \tag{10.12}$$

$$g(\boldsymbol{x}) = \sum_{k=1}^{K_\sigma} \theta_k B_k^{\sigma}(\boldsymbol{x}), \tag{10.13}$$

where $\beta_k, k = 1, \ldots, K_\mu$ and $\theta_k, k = 1, \ldots, K_\sigma$ are the coefficients of the basis functions $B_k^{\mu}(\cdot)$ and $B_k^{\sigma}(\cdot)$, respectively, and K_μ and K_σ are the number of the respective basis functions. According to Denison et al. [47] who propose the Bayesian MARS method, the basis functions should be specified as

$$B_k(\boldsymbol{x}) = \begin{cases} 1, & k = 1, \\ \prod_{j=1}^{J_k} \left[h_{jk} \cdot (x_{r(j,k)} - t_{jk})\right]_+, & k = 2, 3, \ldots, K_\mu \text{ or } K_\sigma. \end{cases} \tag{10.14}$$

Here, $[\cdot]_+ = \max(0, \cdot)$, J_k is the degree of interaction modeled by the basis function $B_k(\boldsymbol{x})$, h_{jk} is the sign indicator, taking the value of either -1 or $+1$, $r(j,k)$ produces the index of the predictor variable which is being split on t_{jk}, whereas t_{jk} is commonly referred to as the knot points.

Lee et al. [131] introduce an integer variable T_k to represent the types of basis functions used in Eq. 10.14. Since two input variables, V and s, are considered for the three inland turbines, there could be three types of basis functions, namely $[\pm(V-*)]_+$ or $[\pm(s-*)]_+$ for the main effect of a respective explanatory variable and $[\pm(V-*)]_+[\pm(s-*)]_+$ for the interactions between them. Let T_k take the integer value of 1, 2, or 3, to represent the three types of basis functions. That is, $[\pm(V-*)]_+$ is represented by $T_k = 1$, $[\pm(s-*)]_+$ represented by $T_k = 2$, and $[\pm(V-*)]_+[\pm(s-*)]_+$ by $T_k = 3$.

To model the location parameter μ for ILT1 and ILT3 data, Lee et al. [131] set $T_k \in \{1,2,3\}$, allowing J_k to take either 1 or 2. For ILT2, however, due to its relatively smaller data amount, a model setting $J_k = 2$ produces unstable and unreasonably wide credible intervals. Consequently, Lee et al. set $T_k \in \{1,2\}$, restricting $J_k = 1$ for ILT2's location parameter μ. For the scale parameter σ, $J_k = 1$ is used for all three datasets. For ITL1 and ILT3, $J_k = 1$ is resulted when setting $T_k \in \{1,2\}$. For ILT2, again due to its data scarcity, Lee et al. include V as the only input variable in the corresponding scale parameter model; this means $T_k = \{1\}$.

Let $\boldsymbol{\Psi}_a = (\boldsymbol{\Psi}_\mu, \boldsymbol{\Psi}_\sigma, \xi)$ denote all the parameters used in the GEV model in Eq. 10.9, where $\boldsymbol{\Psi}_\mu$ and $\boldsymbol{\Psi}_\sigma$ include the parameters in $q(\cdot)$ and $g(\cdot)$, respectively. These parameters are grouped into two sets: (1) the coefficients of the basis functions in $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_{K_\mu})$ or $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_{K_\sigma})$, and (2) the number of knots, the locations of the knots, and the types of basis function in $\boldsymbol{\phi}_\mu$ or $\boldsymbol{\phi}_\sigma$, as follows,

$$\boldsymbol{\phi}_\mu = \left(K_\mu, \boldsymbol{\Lambda}_2^\mu, \ldots, \boldsymbol{\Lambda}_{K_\mu}^\mu\right), \tag{10.15}$$

where

$$\boldsymbol{\Lambda}_k^\mu = \begin{cases} (T_k^\mu, h_{1k}^\mu, t_{1k}^\mu), & \text{when} \quad T_k^\mu = 1,2, \\ (T_k^\mu, h_{1k}^\mu, h_{2k}^\mu, t_{1k}^\mu, t_{2k}^\mu), & \text{when} \quad T_k^\mu = 3, \end{cases}$$

and

$$\boldsymbol{\phi}_\sigma = \left(K_\sigma, \boldsymbol{\Lambda}_2^\sigma, \ldots, \boldsymbol{\Lambda}_{K_\sigma}^\sigma\right), \tag{10.16}$$

where

$$\boldsymbol{\Lambda}_k^\sigma = (T_k^\sigma, h_{1k}^\sigma, t_{1k}^\sigma), \quad \text{when} \quad T_k^\sigma = 1,2.$$

Using the above notations, one can express $\boldsymbol{\Psi}_\mu = (\boldsymbol{\beta}, \boldsymbol{\phi}_\mu)$ and $\boldsymbol{\Psi}_\sigma = (\boldsymbol{\theta}, \boldsymbol{\phi}_\sigma)$.

To complete the Bayesian formulation for the model in Eq. 10.9, priors of the parameters should be specified. Lee et al. [131] use uniform priors on $\boldsymbol{\phi}_\mu$ and $\boldsymbol{\phi}_\sigma$. In the following expressions, we drop the subscript or superscript indicating the association with μ or σ for the sake of notational simplicity,

since the priors for both cases are the same. The following priors are used for variables in $\boldsymbol{\phi}$:

$$
\begin{aligned}
f(K) &= \frac{1}{n}, && K \in \{1, \ldots, n\} \\
f(T_k) &= \begin{cases} 1, & T_k \in \{1\} \quad \text{for } \boldsymbol{\phi}_\sigma \text{ in ILT2,} \\ \frac{1}{2}, & T_k \in \{1,2\} \quad \text{for } \boldsymbol{\phi}_\mu \text{ in ILT2 and all other } \boldsymbol{\phi}'_\sigma s, \\ \frac{1}{3}, & T_k \in \{1,2,3\} \quad \text{for } \boldsymbol{\phi}_\mu \text{ in ILT1 and ILT3,} \end{cases} \\
f(h_{\cdot k}) &= \frac{1}{2}, && h_{\cdot k} \in \{+1, -1\}, \\
f(t_{\cdot k}) &= \frac{1}{n}, && t_{\cdot k} \in \{V_1, \ldots, V_n\} \text{ or } \{s_1, \ldots, s_n\}.
\end{aligned}
$$

In the above, the dot notation in the expressions of $h_{\cdot k}$ and $t_{\cdot k}$ denotes either 1 or 2.

Given $\boldsymbol{\phi}_\mu$ and $\boldsymbol{\phi}_\sigma$, Lee et al. [131] specify the prior distribution for $(\boldsymbol{\beta}, \boldsymbol{\theta}, \xi)$ as the unit information prior (UIP) [118], which is defined by setting the corresponding covariance matrix to be equal to the Fisher information of one observation. This is accomplished by using a multivariate normal prior distribution with its mean set at the maximum likelihood estimate and its covariance matrix as the inverse of the negative of Hessian matrix.

10.4.2 Posterior Distribution of Parameters

The Bayesian MARS model treats the number and locations of the knots as random quantities. When the number of knots changes, the dimension of the parameter space changes with it. To handle a varying dimensionality in the probability distributions in a random sampling procedure, analysts use a reversible jump Markov chain Monte Carlo (RJMCMC) algorithm developed by Green [78]. The acceptance probability for an RJMCMC algorithm includes a Jacobian term, which accounts for the change in dimension. However, under the assumption that the model space for parameters of varying dimension is discrete, there is no need for a Jacobian. In the turbine extreme load analysis, this assumption is satisfied since only are the probable models over possible knot locations and numbers considered. Instead of using the RJMCMC algorithm, Lee et al. [131] use the reversible jump sampler (RJS) algorithm proposed in [46]. Because the RJS algorithm does not require new parameters to match dimensions between models nor the corresponding Jacobian term in the acceptance probability, it is simpler and more efficient to execute.

To allow for dimensional changes, there are three actions in an RJS algorithm [46, page 53]: BIRTH, DEATH and MOVE, which adds, deletes, or alters a basis function, respectively. Accordingly, the number of knots as well as the locations of some knots change. Denison et al. [46] suggest using equal probability, i.e., 1/3, to propose any of the three moves, and then, use the following acceptance probability, α, while executing a proposed move from a

model having k basis functions to a model having k^c basis functions:

$$\alpha = \min\{1, \text{the ratio of marginal likelihood} \times \mathcal{R}\}, \tag{10.17}$$

where $\mathcal{R}$ is a ratio of probabilities defined as:

- For a BIRTH action, $\mathcal{R} = \dfrac{\text{probability of DEATH in model } k^c}{\text{probability of BIRTH in model } k}$;
- For a DEATH action, $\mathcal{R} = \dfrac{\text{probability of BIRTH in model } k^c}{\text{probability of DEATH in model } k}$;
- For a MOVE action, $\mathcal{R} = \dfrac{\text{probability of MOVE in model } k^c}{\text{probability of MOVE in model } k}$.

Lee et al. [131] state that they have $\mathcal{R} = 1$ for most cases, because the probabilities in the denominator and numerator are equal, except when k reaches either the upper or the lower bound.

The marginal likelihood in Eq. 10.17 is expressed as

$$f(\mathcal{D}_z|\boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma) = \int f(\mathcal{D}_z|\boldsymbol{\beta}, \boldsymbol{\theta}, \xi, \boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma)\, f(\boldsymbol{\beta}, \boldsymbol{\theta}, \xi|\boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma)\, d\boldsymbol{\beta} d\boldsymbol{\theta} d\xi, \tag{10.18}$$

where $\mathcal{D}_z = (z_1, \ldots, z_n)$ represents a set of observed load data. Since it is difficult to calculate the above marginal likelihood analytically, Lee et al. [131] consider an approximation of $f(\mathcal{D}_z|\boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma)$. Kass and Wasserman [118] and Raftery [171] show that when UIPs are used, the marginal log-likelihood, i.e., $\log(f(\mathcal{D}_z|\boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma))$, can be reasonably approximated by the Schwarz information criterion (SIC) [197], also known as BIC; please refer to Eq. 2.23.

The SIC is expressed as

$$\text{SIC}_{\boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma} = \log\left(f(\mathcal{D}_z|\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}, \hat{\xi}, \boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma)\right) - \frac{1}{2} d_k \log(n), \tag{10.19}$$

where $\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\theta}}, \hat{\xi}$ are the MLEs of the corresponding parameters obtained conditional on $\boldsymbol{\phi}_\mu$ and $\boldsymbol{\phi}_\sigma$, and d_k is the total number of parameters to be estimated. In this case, $d_k = K_\mu + K_\sigma + 1$ (the inclusion of the last 1 is due to ξ).

Comparing Eq. 10.19 with Eq. 2.23, one may notice that the two expressions are indeed equivalent but differ by a constant of -2. Note that in Chapter 2, a smaller BIC implies a better model fit to data. Here, a larger SIC suggests a better model fit, because of this -2 difference.

There are two dimension-varying states, $\boldsymbol{\phi}_\mu$ and $\boldsymbol{\phi}_\sigma$, in the RJS algorithm. Consequently, two marginal log-likelihood ratios are needed. They are approximated by the corresponding SICs, such as

$$\log \frac{f(\mathcal{D}_z|\boldsymbol{\phi}_\mu^c, \boldsymbol{\phi}_\sigma)}{f(\mathcal{D}_z|\boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma)} \backsimeq \text{SIC}_{\boldsymbol{\phi}_\mu^c, \boldsymbol{\phi}_\sigma} - \text{SIC}_{\boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma}, \tag{10.20}$$

and

$$\log \frac{f(\mathcal{D}_z|\boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma^c)}{f(\mathcal{D}_z|\boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma)} \backsimeq \text{SIC}_{\boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma^c} - \text{SIC}_{\boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma}. \tag{10.21}$$

Then, one uses two acceptance probabilities α_μ and α_σ for accepting or rejecting a new state in $\boldsymbol{\phi}_\mu$ and $\boldsymbol{\phi}_\sigma$, respectively. Using the SICs, α_μ and α_σ are expressed as:

$$\alpha_\mu = \min\left\{1, \exp\left(\mathrm{SIC}_{\boldsymbol{\phi}_\mu^c, \boldsymbol{\phi}_\sigma} - \mathrm{SIC}_{\boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma}\right) \times \mathcal{R}\right\}, \tag{10.22}$$

and

$$\alpha_\sigma = \min\left\{1, \exp\left(\mathrm{SIC}_{\boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma^c} - \mathrm{SIC}_{\boldsymbol{\phi}_\mu, \boldsymbol{\phi}_\sigma}\right) \times \mathcal{R}\right\}. \tag{10.23}$$

In order to produce the samples from the posterior distribution of parameters in $\boldsymbol{\Psi}_a$, Lee et al. [131] sequentially draw samples for $\boldsymbol{\phi}_\mu$ and $\boldsymbol{\phi}_\sigma$ by using the two acceptance probabilities while marginalizing out ($\boldsymbol{\beta}$, $\boldsymbol{\theta}$, ξ), and then, conditional on the sampled $\boldsymbol{\phi}_\mu$ and $\boldsymbol{\phi}_\sigma$, draw samples for ($\boldsymbol{\beta}$, $\boldsymbol{\theta}$, ξ) using a normal approximation based on the maximum likelihood estimates and the observed information matrix.

10.4.3 Wind Characteristics Model

To find a site-specific load distribution, the distribution of wind characteristics $f(\boldsymbol{x})$ in Eq. 10.4 needs to be specified. Since a statistical correlation is noticed in Fig. 10.4 between the 10-minute average wind speed, V, and the standard deviation of wind speed, s, the distribution of wind characteristics $f(\boldsymbol{x})$ can be written as a product of the average wind speed distribution $f(V)$ and the conditional wind standard deviation distribution $f(s|V)$.

The probabilistic distribution of wind speed, $f(V)$, is discussed in Chapter 2. At that time, the discussion concentrates on Weibull distribution. The three-parameter Weibull distribution fits the three wind turbine datasets well, as one will see in Section 10.6.1, and is in fact the one used in the case study.

For modeling the 10-minute average wind speed V, the IEC standard suggests using a two-parameter Weibull distribution (W2) or a Rayleigh distribution (RAY) [101]. These two distributions are arguably the most widely used ones for this purpose. But analysts [31, 134] note that under different wind regimes other distributions may fit wind speed data better, including the three-parameter Weibull distribution (W3), three-parameter log-normal distribution (LN3), three-parameter Gamma distribution (G3), and three-parameter inverse-Gaussian distribution (IG3).

What Lee et al. [131] suggest to do is to take the total of six candidate distribution models for average wind speed (W2, W3, RAY, LN3, G3, IG3) and conduct a Bayesian model selection to choose the best distribution fitting a given average wind speed dataset. Lee et al. assume UIP for the parameters involved in the aforementioned models, and as such, the Bayesian model selection is again based on maximizing the SIC. The chosen best wind speed model is denoted by $\mathcal{M}_V$. Then, the distribution of 10-minute average wind speed V is expressed as

$$V_i \sim \mathcal{M}_V(\boldsymbol{\nu}), \tag{10.24}$$

where $\boldsymbol{\nu}$ is the set of parameters specifying $\mathcal{M}_V$. For instance, if $\mathcal{M}_V$ is W3,

then $\boldsymbol{\nu} = (\nu_1, \nu_2, \nu_3)$, where ν_1, ν_2, and ν_3 represent, respectively, the location, scale, and shape parameter of the three-parameter Weibull distribution.

For modeling the standard deviation of wind speed s, given the average wind speed V, the IEC standard [101] recommends using a two-parameter truncated normal distribution (TN2), which appears to be what analysts have commonly used [63]. The distribution is characterized by a location parameter η and a scale parameter δ. In the literature, both η and δ are treated as a constant. But Lee et al. [131] observe that datasets measured at different sites have different relationships between the average wind speed V and the standard deviation s. Some of the V-versus-s scatter plots show nonlinear patterns.

Motivated by this observation, Lee et al. [131] employ a Bayesian MARS model for modeling η and δ, similar to what is done in Section 10.4.1 for the conditional load model. The standard deviation of wind speed s, conditional on the average wind speed V, can then be expressed as

$$s_i|V_i \sim \text{TN2}(\eta(V_i), \delta(V_i)), \tag{10.25}$$

where $\eta(V_i) = q_\eta(V_i)$ and $\delta(V_i) = \exp(g_\delta(V_i))$, like their counterparts in Eq. 10.10 to Eq. 10.13, are linear combinations of the basis functions taking the general form as in Eq. 10.14. Notice that both of the functions have only one input variable, which is the average wind speed.

Let $\boldsymbol{\Psi}_\eta = (\boldsymbol{\beta}_\eta, \boldsymbol{\phi}_\eta)$ and $\boldsymbol{\Psi}_\delta = (\boldsymbol{\theta}_\delta, \boldsymbol{\phi}_\delta)$ denote the parameters in $q_\eta(\cdot)$ and $g_\delta(\cdot)$. Since the basis functions for q_η and g_δ have a single input variable, only one type of basis function is needed, i.e., $T_k = 1$. For this reason, $\boldsymbol{\phi}_\eta$ and $\boldsymbol{\phi}_\delta$ are much simpler than $\boldsymbol{\phi}_\mu$ and $\boldsymbol{\phi}_\sigma$, their counterparts in Eq. 10.15 and Eq. 10.16, and are expressed as follows:

$$\begin{aligned} \boldsymbol{\phi}_\eta &= \left(K_\eta, \boldsymbol{\Lambda}_2^\eta, \ldots, \boldsymbol{\Lambda}_{K_\eta}^\eta\right), \\ &\quad \text{where } \boldsymbol{\Lambda}_k^\eta = (T_k^\eta, h_{1k}^\eta, t_{1k}^\eta) \quad \text{and} \quad T_k^\eta = 1; \end{aligned} \tag{10.26}$$

and

$$\begin{aligned} \boldsymbol{\phi}_\delta &= \left(K_\delta, \boldsymbol{\Lambda}_2^\delta, \ldots, \boldsymbol{\Lambda}_{K_\delta}^\delta\right), \\ &\quad \text{where } \boldsymbol{\Lambda}_k^\delta = (T_k^\delta, h_{1k}^\delta, t_{1k}^\delta) \quad \text{and} \quad T_k^\delta = 1. \end{aligned} \tag{10.27}$$

Lee et al. [131] choose the prior distribution for $(\boldsymbol{\beta}_\eta, \boldsymbol{\theta}_\delta)$ as UIP, the prior for $(\boldsymbol{\phi}_\eta, \boldsymbol{\phi}_\delta)$ as uniform distribution, and set $f(T_k) = 1$ because T_k is always 1. They solve this Bayesian MARS model using an RJS algorithm, as in the preceding two sections.

The predictive distributions of the average wind speed $\tilde{V}$ and the standard deviation $\tilde{s}$ are

$$f(\tilde{V}|\mathcal{D}_V) = \int f(\tilde{V}|\boldsymbol{\nu}, \mathcal{D}_V) f(\boldsymbol{\nu}|\mathcal{D}_V) d\boldsymbol{\nu}, \tag{10.28}$$

and

$$f(\tilde{s}|\tilde{V}, \mathcal{D}_V, \mathcal{D}_s) = \int\int f(\tilde{s}|\tilde{V}, \boldsymbol{\Psi}_\eta, \boldsymbol{\Psi}_\delta, \mathcal{D}_V, \mathcal{D}_s) f(\boldsymbol{\Psi}_\eta, \boldsymbol{\Psi}_\delta|\mathcal{D}_V, \mathcal{D}_s) d\boldsymbol{\Psi}_\eta d\boldsymbol{\Psi}_\delta. \tag{10.29}$$

10.4.4 Posterior Predictive Distribution

Analysts are interested in getting the posterior predictive distribution of the quantile value l_T, based on the observed load and wind data $\mathcal{D} := (\mathcal{D}_z, \mathcal{D}_V, \mathcal{D}_s)$. Under a Bayesian framework, one draws samples, $\tilde{z}$'s, from the predictive distribution of the maximum load, $f(\tilde{z}|\mathcal{D}, \boldsymbol{\Psi}_a)$, which is

$$f(\tilde{z}|\mathcal{D}, \boldsymbol{\Psi}_a) = \int\int f(\tilde{z}|\tilde{V}, \tilde{s}, \boldsymbol{\Psi}_a, \mathcal{D}) f(\tilde{V}, \tilde{s}|\mathcal{D}_V, \mathcal{D}_s) d\tilde{V} d\tilde{s}, \quad (10.30)$$

where $f(\tilde{V}, \tilde{s}|\mathcal{D}_V, \mathcal{D}_s)$ can be expressed as the product of Eq. 10.28 and Eq. 10.29.

To calculate a quantile value of the load for a given P_T, one goes through the steps in Algorithm 10.2. The predictive mean and Bayesian credible interval of the extreme load level, l_T, are obtained when running the RJS algorithm. The RJS runs through M_l iterations, and at each iteration, one obtains a set of samples of the model parameters, $\boldsymbol{\Psi}_a$, and calculates an $l_T[\boldsymbol{\Psi}_a]$. Once the M_l values of $l_T[\boldsymbol{\Psi}_a]$ are obtained, the mean and credible interval of l_T can then be numerically computed.

Algorithm 10.2 Sampling procedure to obtain the posterior predictive distribution of load response z. Set $M_w = 1,000$, $M_l = 10,000$, $N_w = 100$, and $N_l = 100$.

---

1. Draw $M_w \times N_w$ samples from the joint posterior predictive distribution $f\left(\tilde{V}, \tilde{s}|\mathcal{D}_V, \mathcal{D}_s\right)$ of wind characteristics $(\tilde{V}, \tilde{s})$. This is realized by employing Algorithm 10.4;

2. Draw a set of samples from the posterior distribution of model parameters $\boldsymbol{\Psi}_a = (\boldsymbol{\Psi}_\mu, \boldsymbol{\Psi}_\sigma, \xi)$. This is realized by employing the RJS algorithm in Section 10.4.2 (or Steps 1–11 of Algorithm 10.3);

3. Given the above samples of wind characteristics and model parameters, one calculates (μ, σ, ξ) that are needed in a GEV distribution. This yields a short-term distribution $f\left(\tilde{z}|\tilde{V}, \tilde{s}, \boldsymbol{\Psi}_a\right)$;

4. Integrate out the wind characteristics $(\tilde{V}, \tilde{s})$, as implied in Eq. 10.30, to obtain the long-term distribution $f(\tilde{z}|\mathcal{D}, \boldsymbol{\Psi}_a)$.

5. Draw $N_l \times M_w \times N_w$ samples from $f(\tilde{z}|\mathcal{D}, \boldsymbol{\Psi}_a)$ and compute the quantile value $l_T[\boldsymbol{\Psi}_a]$ corresponding to P_T.

6. Repeat the above procedure M_l times to get the median and confidence interval of l_T.

10.5 ALGORITHMS USED IN BAYESIAN INFERENCE

In this section, more details of the implementation procedures are provided to facilitate the Bayesian inference.

The procedure consists of two main parts: Algorithm 10.3, which is to construct the posterior predictive distribution of the extreme load level l_T, and Algorithm 10.4, which is to obtain the posterior predictive distribution of wind characteristics (V, s). The main algorithms use the RJS subroutine for the location parameter μ and the scale parameter σ. These two subroutines are separately listed in Algorithms 10.5 and 10.6. The two subroutines look the same but differ in terms of the specific variables and parameters used therein.

Algorithm 10.2 in Section 10.4.4 carries out the same task as Algorithm 10.3 does. The difference is that Algorithm 10.2 outlines the main steps, whereas Algorithm 10.3 presents more detailed steps.

10.6 CASE STUDY

This section presents numerical analysis of extreme loads recorded in the `Turbine Bending Moment Dataset` and discusses the difference between the spline-based approach and the binning-based approach.

10.6.1 Selection of Wind Speed Model

The first task is to select a model, out of the six candidate models mentioned in Section 10.4.3, for the average wind speed. This model selection is done using the SIC.

Table 10.2 presents the SIC values of the six candidate average wind speed models using a respective ILT dataset. The boldfaced values indicate the largest SIC for a given dataset, and accordingly, the corresponding model is chosen for that dataset.

Regarding the average wind speed model, all candidate distributions except RAY provide generally a good model fit for ILT1 with a similar level of fitting quality, but W3 outperforms others slightly. For the ILT2 data, W2, W3, LN3 and G3 produce similar SIC values. In the ILT3 data, W3, LN3, G3 and IG3 perform similarly. Again W3 is slightly better. For this reason, W3 is chosen as the average wind speed model.

10.6.2 Pointwise Credible Intervals

As a form of checking the conditional maximum load model, Lee et al. [131] produce the 95% pointwise credible intervals of the load response under different wind speeds and standard deviations. The resulting credible intervals are presented in Figs. 10.6 and 10.7.

To generate these figures, Lee et al. [131] take a dataset and fix V or s at one specific speed or standard deviation at a time and then draw the posterior

Algorithm 10.3 Construct the posterior predictive distribution of the extreme load level using the Bayesian spline models. Set $M_w = 1,000$, $M_l = 10,000$, $N_w = 100$, and $N_l = 100$.

1. Set $t = 0$ and the initial $\boldsymbol{\phi}_\mu^{(t)}$ and $\boldsymbol{\phi}_\sigma^{(t)}$ both to be a constant scalar.
2. At iteration t, K_μ and K_σ are equal to the number of basis functions specified in $\boldsymbol{\phi}_\mu^{(t)}$ and $\boldsymbol{\phi}_\sigma^{(t)}$. Find the MLEs of $\boldsymbol{\beta}^{(t)}, \boldsymbol{\theta}^{(t)}, \xi^{(t)}$ and the inverse of the negative of Hessian matrix, given $\boldsymbol{\phi}_\mu^{(t)}$ and $\boldsymbol{\phi}_\sigma^{(t)}$.
3. Generate u_μ^1 uniformly on $[0, 1]$ and choose a move in the RJS procedure. Denote by $b_{K_\mu}, r_{K_\mu}, m_{K_\mu}$ the proposal probabilities associated with a move type; they are all set as $\frac{1}{3}$. Call Algorithm 10.5 to execute the RJS procedure.
4. Find the MLEs $(\boldsymbol{\beta}^*, \boldsymbol{\theta}^*, \xi^*)$ and the inverse of the negative of Hessian matrix, given $\boldsymbol{\phi}_\mu^*$ and $\boldsymbol{\phi}_\sigma^{(t)}$.
5. Generate u_μ^2 uniformly on $[0, 1]$ and compute the acceptance ratio α_μ in Eq. 10.22, using the results from Step 2 and Step 4.
6. Accept $\boldsymbol{\phi}_\mu^*$ as $\boldsymbol{\phi}_\mu^{(t+1)}$ with probability $\min(\alpha_\mu, 1)$. If $\boldsymbol{\phi}_\mu^*$ is not accepted, let $\boldsymbol{\phi}_\mu^{(t+1)} = \boldsymbol{\phi}_\mu^{(t)}$.
7. Generate u_σ^1 uniformly on $[0, 1]$ and choose a move in the RJS procedure. Denote by $b_{K_\sigma}, r_{K_\sigma}, m_{K_\sigma}$ the proposal probabilities associated with a move type; they are all set as $\frac{1}{3}$. Call Algorithm 10.6 to execute the RJS procedure.
8. Find the MLEs $(\boldsymbol{\beta}^*, \boldsymbol{\theta}^*, \xi^*)$ and the inverse of the negative of Hessian matrix, given $\boldsymbol{\phi}_\mu^{(t+1)}$ and $\boldsymbol{\phi}_\sigma^*$.
9. Generate u_σ^2 uniformly on $[0, 1]$ and compute the acceptance ratio α_σ in Eq. 10.23, using the results from Step 4 and Step 8.
10. Accept $\boldsymbol{\phi}_\sigma^*$ as $\boldsymbol{\phi}_\sigma^{(t+1)}$ with probability $\min(\alpha_\sigma, 1)$. If $\boldsymbol{\phi}_\sigma^*$ is not accepted, let $\boldsymbol{\phi}_\sigma^{(t+1)} = \boldsymbol{\phi}_\sigma^{(t)}$.
11. After initial burn-ins (set to 1,000 samples), draw a posterior sample of $(\boldsymbol{\beta}^{(t+1)}, \boldsymbol{\theta}^{(t+1)}, \xi^{(t+1)})$ from the approximated multivariate normal distribution at the maximum likelihood estimates and the inverse of the negative of Hessian matrix. Depending on the acceptance or rejection that happened in Step 6 and Step 10, the MLEs to be used are obtained from either Step 2, Step 4, or Step 8.
12. Take the posterior sample of $\boldsymbol{\Psi}_a$, obtained in Step 6, Step 10, and Step 11, and calculate a sample of μ and σ using Eq. 10.10 and Eq. 10.11, respectively, for each pair of the $M_w \times N_w$ samples of (V, s) obtained in Algorithm 10.4. This generates $M_w \times N_w$ samples of μ and σ.
13. Draw N_l samples for the 10-minute maximum load $\tilde{z}$ from each GEV distribution with μ_i, σ_i, and ξ_i, $i = 1, \ldots, M_w \times N_w$, where μ_i and σ_i are among $M_w \times N_w$ samples obtained in Step 12, and ξ_i is always set as $\xi^{(t+1)}$.
14. Get the quantile value (that is, the extreme load level $l_T[\boldsymbol{\Psi}_a]$) corresponding to $1 - P_T$ from the $M_w \times N_w \times N_l$ samples of $\tilde{z}$.
15. To obtain a credible interval for l_T, repeat Step 2 through Step 14 M_l times.

Algorithm 10.4 Obtain the posterior predictive distribution of wind characteristics (V, s). Set $M_w = 1,000$ and $N_w = 100$.

1. Find the MLEs of $\boldsymbol{\nu}$ for all candidate distributions listed in Section 10.4.3.
2. Use the SIC to select the "best" distribution model for the average wind speed V. The chosen distribution model is used in the subsequent steps to draw posterior samples.
3. Draw a posterior sample of $\boldsymbol{\nu}$ from the approximated multivariate normal distribution at the MLEs and the inverse of the negative of Hessian matrix.
4. Draw N_w samples of $\tilde{V}$ using the distribution chosen in Step 2 with the parameter sampled in Step 3.
5. Implement the RJS algorithm again, namely Step 1 through Step 11 in Algorithm 10.3, to get one posterior sample of $\boldsymbol{\Psi}_\eta = (\boldsymbol{\beta}_\eta, \boldsymbol{\phi}_\eta)$ and $\boldsymbol{\Psi}_\delta = (\boldsymbol{\theta}_\delta, \boldsymbol{\phi}_\delta)$.
6. Take the posterior sample of $\boldsymbol{\Psi}_\eta$ and $\boldsymbol{\Psi}_\delta$, obtained in Step 5, and calculate a sample of η and δ using Eq. 10.25 for each sample of $\tilde{V}$. This generates N_w samples of η and δ.
7. Draw a sample for the standard deviation of wind speed $\tilde{s}$ from each truncated normal distribution with η_i, δ_i, $i = 1, \ldots, N_w$. Using the N_w samples of η and δ obtained in Step 6, one obtains N_w samples of $\tilde{s}$.
8. To get $M_w \times N_w$ samples of $\tilde{V}$ and $\tilde{s}$, repeat Step 3 through Step 7 M_w times.

Algorithm 10.5 Three types of move in the RJS for location parameter μ.

1. If $u_\mu^1 \leq b_{K_\mu}$, then go to BIRTH step, denoted by $\boldsymbol{\phi}_\mu^* = \text{BIRTH-proposal}(\boldsymbol{\phi}_\mu^{(t)})$, which is to augment $\boldsymbol{\phi}_\mu^{(t)}$ with a $\boldsymbol{\Lambda}_{K_\mu+1}^\mu$ that is selected uniformly at random;
2. Else if $b_{K_\mu} \leq u_\mu^1 \leq b_{K_\mu} + r_{K_\mu}$,
   then go to DEATH step, denoted by $\boldsymbol{\phi}_\mu^* = \text{DEATH-proposal}(\boldsymbol{\phi}_\mu^{(t)})$, which is to remove from $\boldsymbol{\phi}_\mu^{(t)}$ with a $\boldsymbol{\Lambda}_k^\mu$ where $2 \leq k \leq K_\mu$ that is selected uniformly at random;
3. Else go to MOVE step, denoted by $\boldsymbol{\phi}_\mu^* = \text{MOVE-proposal}(\boldsymbol{\phi}_\mu^{(t)})$, which first do $\boldsymbol{\phi}_\mu^\dagger = \text{DEATH-proposal}(\boldsymbol{\phi}_\mu^{(t)})$ and then do $\boldsymbol{\phi}_\mu^* = \text{BIRTH-proposal}(\boldsymbol{\phi}_\mu^\dagger)$.

Algorithm 10.6 Three types of move in the RJS for scale parameter σ.

1. If $u_\sigma^1 \leq b_{K_\sigma}$, then go to BIRTH step, denoted by $\boldsymbol{\phi}_\sigma^*$ = BIRTH-proposal($\boldsymbol{\phi}_\sigma^{(t)}$), which is to augment $\boldsymbol{\phi}_\sigma^{(t)}$ with a $\boldsymbol{\Lambda}_{K_\sigma+1}^\sigma$ that is selected uniformly at random;

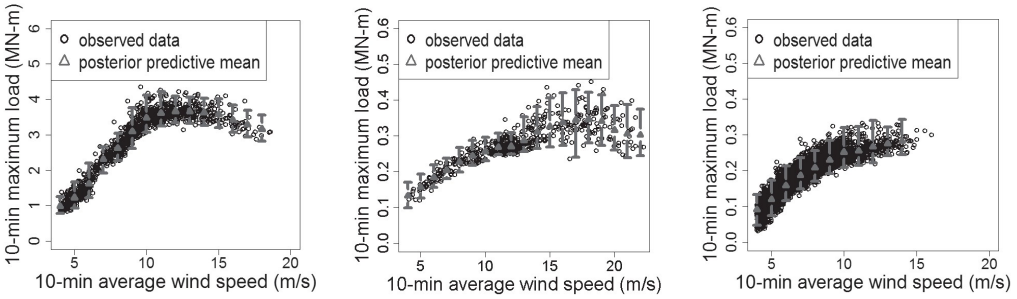
2. Else if $b_{K_\sigma} \leq u_\sigma^1 \leq b_{K_\sigma} + r_{K_\sigma}$,
then go to DEATH step, denoted by $\boldsymbol{\phi}_\sigma^*$ = DEATH-proposal($\boldsymbol{\phi}_\sigma^{(t)}$), which is to remove from $\boldsymbol{\phi}^{(t)}$ with a $\boldsymbol{\Lambda}_k^\sigma$ where $2 \leq k \leq K_\sigma$ that is selected uniformly at random;

3. Else go to MOVE step, denoted by $\boldsymbol{\phi}_\sigma^*$ =MOVE-proposal($\boldsymbol{\phi}_\sigma^{(t)}$), which first do $\boldsymbol{\phi}_\sigma^\dagger$ = DEATH-proposal($\boldsymbol{\phi}_\sigma^{(t)}$) and then do $\boldsymbol{\phi}_\sigma^*$ = BIRTH-proposal($\boldsymbol{\phi}_\sigma^\dagger$).

TABLE 10.2 SIC for the average wind speed models.

Distributions	ILT1	ILT2	ILT3
W2	-2,984	-1,667	-12,287
W3	**-2, 941**	**-1, 663**	**-11, 242**
RAY	-3,120	-1,779	-13,396
LN3	-2,989	-1,666	-11,444
G3	-2,974	-1,666	-11,290
IG3	-2,986	-2,313	-11,410

Source: Lee et al. [131]. With permission.



(a) ILT1 (b) ILT2 (c) ILT3

FIGURE 10.6 The 95% pointwise credible intervals of the load response against wind speeds. (Reprinted with permission from Lee et al. [131].)

samples for $\tilde{z}$ from the posterior predictive distribution of conditional maximum load, $f(\tilde{z}|\boldsymbol{x})$. Suppose that one wants to generate the credible interval at wind speed V_* or standard deviation s_*. The posterior predictive distributions are computed as follows:

$$f(\tilde{z}|(V,s) \in \mathcal{D}_{V_*}, \mathcal{D}_z) = \int f(\tilde{z}|(V,s) \in \mathcal{D}_{V_*}, \boldsymbol{\Psi}_a) f(\boldsymbol{\Psi}_a|\mathcal{D}_z) d\boldsymbol{\Psi}_a, \tag{10.31}$$

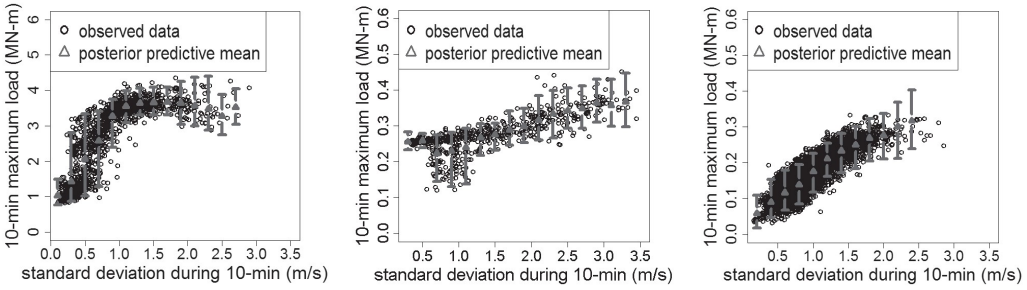
and

$$f(\tilde{z}|(V,s) \in \mathcal{D}_{s_*}, \mathcal{D}_z) = \int f(\tilde{z}|(V,s) \in \mathcal{D}_{s_*}, \boldsymbol{\Psi}_a) f(\boldsymbol{\Psi}_a|\mathcal{D}_z) d\boldsymbol{\Psi}_a, \tag{10.32}$$

where $\mathcal{D}_{V_*}$ and $\mathcal{D}_{s_*}$ are subsets of the observed data such that $\mathcal{D}_{V_*} = \{(V_i, s_i) : V_* - 0.5 < V_i < V_* + 0.5, \text{ and, } (V_i, s_i) \in \mathcal{D}_{V,s}\}$ and $\mathcal{D}_{s_*} = \{(V_i, s_i) : s_* - 0.05 < s_i < s_* + 0.05, \text{ and, } (V_i, s_i) \in \mathcal{D}_{V,s}\}$. Given these distributions, samples for $\tilde{z}$ are drawn to construct the 95% credible interval at V_* or s_*. The result is shown as one vertical bar in either a V-plot in Fig. 10.6 or an s-plot in Fig. 10.7. To complete these figures, the process is repeated in the V-domain with 1 m/s increment and in the s-domain with 0.2 m/s increment. These figures show that the variability in data are reasonably captured by the spline method.

10.6.3 Binning versus Spline Methods

In the procedure of estimating the extreme load level, two different distributions of maximum load z are involved—one is the conditional maximum load distribution $f(z|\boldsymbol{x})$, namely the short-term distribution, and the other is the unconditional maximum load distribution $f(z)$, namely the long-term distribution. Using the observed field data, it is difficult to assess the estimation accuracy of the extreme load levels in the long-term distribution,



(a) ILT1 (b) ILT2 (c) ILT3

FIGURE 10.7 The 95% pointwise credible intervals of the load response against standard deviations. (Reprinted with permission from Lee et al. [131].)

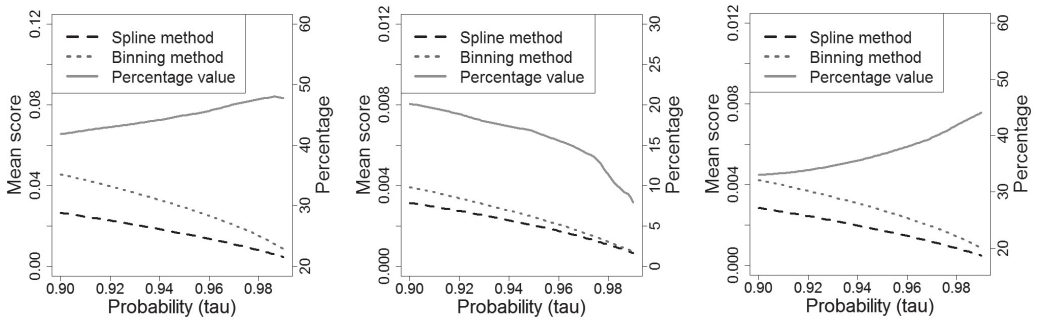
because of the relatively small amount of observation records. For this reason, this section evaluates a method's performance of estimating the tail of the short-term distribution $f(z|\boldsymbol{x})$. Doing so makes sense, as the short-term distribution underlies the difference between the Bayesian spline method and the IEC standard procedure based on binning. In Section 10.6.5, a simulation is employed to generate a much larger dataset, allowing to compare the performance of the two methods in estimating the extreme load level in the long-term distribution.

To evaluate the tail part of a conditional maximum load distribution, Lee et al. [131] compute a set of upper quantile estimators and assess their estimation qualities using the generalized piecewise linear (GPL) loss function [73]. A GPL is defined as follows:

$$S_{\tau,b}(\hat{l}(\boldsymbol{x}_i), z(\boldsymbol{x}_i)) = \begin{cases} \left(\mathbb{1}(\hat{l}(\boldsymbol{x}_i) \geq z(\boldsymbol{x}_i)) - \tau\right) \frac{1}{|b|}([\hat{l}(\boldsymbol{x}_i)]^b - [z(\boldsymbol{x}_i)]^b), & \text{for} \quad b \neq 0, \\ \left(\mathbb{1}(\hat{l}(\boldsymbol{x}_i) \geq z(\boldsymbol{x}_i)) - \tau\right) \log\left(\frac{\hat{l}(\boldsymbol{x}_i)}{z(\boldsymbol{x}_i)}\right), & \text{for} \quad b = 0, \end{cases} \tag{10.33}$$

where $\hat{l}(\boldsymbol{x}_i)$ is the τ-quantile estimation of $f(z|\boldsymbol{x}_i)$ for a given $\boldsymbol{x}_i$, $z(\boldsymbol{x}_i)$ is the observed maximum load in the test dataset, given the same $\boldsymbol{x}_i$, b is a power parameter, and $\mathbb{1}$ is the indicator function. The power parameter b usually ranges between 0 and 2.5. When $b = 1$, the GPL loss function is the same as the piecewise linear (PL) loss function.

For the above empirical evaluation, Lee et al. [131] randomly divide a dataset into a partition of 80% for training and 20% for testing. They use the training set to establish a short-term distribution $f(z|\boldsymbol{x})$. For any $\boldsymbol{x}_i$ in the test set, the τ-quantile estimation $\hat{l}(\boldsymbol{x}_i)$ can be computed using $f(z|\boldsymbol{x})$. And then, the GPL loss function value is taken as the average of all $S_{\tau,b}$ values



(a) ILT1 (b) ILT2 (c) ILT3

FIGURE 10.8 Comparison of piecewise linear loss function: the left vertical axis represents the mean score values and the right vertical axis represents a percentage value, which is the reduction in the mean scores when the spline method is compared with the binning method. (Reprinted with permission from Lee et al. [131].)

over the test set, as follows:

$$\overline{S}_{\tau,b} = \frac{1}{n_t} \sum_{i=1}^{n_t} S_{\tau,b}(\hat{l}(\boldsymbol{x}_i), z_i), \tag{10.34}$$

where n_t is the number of data points in a test set, and z_i is the same as $z(\boldsymbol{x}_i)$. Apparently, $\overline{S}_{\tau,b}$ is a mean score. The training/test procedure is repeated 10 times, and the final mean score is the average of the ten mean scores. For notational simplicity, this final mean score is still called the mean score and represented by $\overline{S}_{\tau,b}$, as long as its meaning is clear in the context.

In this comparison, two methods are used to establish the short-term distribution: the binning method and the Bayesian spline method. In the sampling algorithms outlined in Sections 10.3 and 10.5, $N_l = 100$ samples are drawn from the short-term distribution. As such, one can evaluate the quality of quantile estimations of the short-term distribution for a τ up to 0.99.

First, let us take a look at the comparisons in Fig. 10.8, which compares the PL loss (i.e., $b = 1$) of both methods as τ varies in the above-mentioned range. The left vertical axis shows the values of the mean score of the PL loss, whereas the right axis is the percentage value of the reduction in mean scores when the spline method is compared with the binning method. For all three datasets, the spline method maintains lower mean scores than the binning method.

When τ is approaching 0.99 in Fig. 10.8, it looks like that the PL losses of the spline and binning methods are getting closer to each other. This is largely due to the fact that the PL loss values are smaller at a higher τ, so that their differences are compressed in the figure. If one looks at the solid line in the plots, which represents the percentage of reduction in the mean

TABLE 10.3 Mean scores of GPL/PL for the 0.9-quantile estimators.

Power parameter	ILT1		ILT2		ILT3	
	Binning	Spline	Binning	Spline	Binning	Spline
$b = 0$	0.0185	0.0108	0.0129	0.0103	0.0256	0.0171
$b = 1$	0.0455	0.0265	0.0040	0.0031	0.0042	0.0028
$b = 2$	0.1318	0.0782	0.0013	0.0010	0.0008	0.0005

Source: Lee et al. [131]. With permission.

TABLE 10.4 Mean scores of GPL/PL for the 0.99-quantile estimators.

Power parameter	ILT1		ILT2		ILT3	
	Binning	Spline	Binning	Spline	Binning	Spline
$b = 0$	0.0031	0.0018	0.0022	0.0020	0.0045	0.0027
$b = 1$	0.0086	0.0045	0.0007	0.0006	0.0008	0.0005
$b = 2$	0.0270	0.0135	0.0003	0.0002	0.0002	0.0001

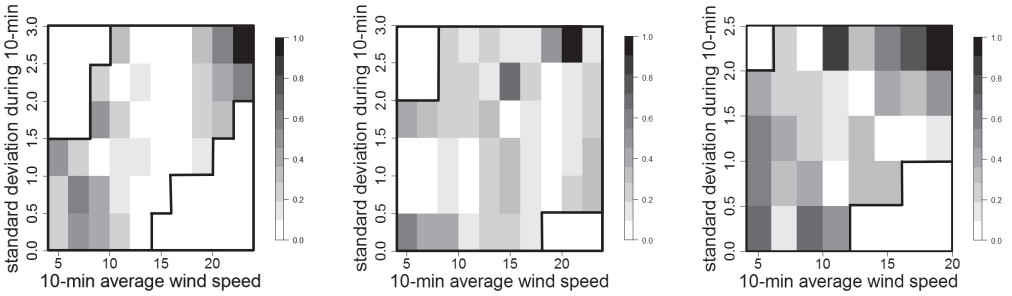
Source: Lee et al. [131]. With permission.

score, the spline method's advantage over the binning method is more evident in the cases of ILT1 and ILT3 datasets. When τ gets larger, the spline method produces a significant improvement over the binning method, with a reduction of PL loss ranging from 33% to 50%. The trend is different when using the ILT2 dataset. But still, the spline method can reduce the mean scores of the PL loss from the binning method by 8% to 20%. Please note that ILT2 dataset is the smallest set, having slightly fewer than 600 data records. The difference observed over the ILT2 case is likely attributable to the scarcity of data.

Lee et al. [131] compute the mean scores of the GPL loss under three different power parameters $b = 0, 1, 2$ for each method. Table 10.3 presents the results under $\tau = 0.9$, whereas Table 10.4 is for $\tau = 0.99$. In Table 10.3, the spline method has a mean score 20% to 42% lower than the binning method. In Table 10.4, the reductions in mean scores are in a similar range.

In order to understand the difference between the spline method and binning method, Lee et al. [131] compare the 0.99 quantiles of the 10-minute maximum load conditional on a specific wind condition. This is done by computing the difference in the quantile values of conditional maximum load from the two methods for different weather bins. The wind condition of each bin is approximated by the median values of V and s in that bin. Fig. 10.9 shows the standardized difference of the two 0.99 quantile values in each bin. The darker the color is, the bigger the difference. Lee et al. exclude comparisons in the weather bins with very low likelihood, which is the bins of low wind speed and high standard deviation or high wind speed and low standard deviation.

One can observe that the two methods produce similar results at the bins having a sufficient number of data points, which are mostly weather bins in the central area. The results are different when the data are scarce—this tends to



(a) ILT1 (b) ILT2 (c) ILT3

FIGURE 10.9 Comparison of the 0.99-quantiles between binning method and spline method. (Reprinted with permission from Lee et al. [131].)

TABLE 10.5 Estimates of extreme load levels ($l_T, T = 20$ years), unit: MN-m.

Datasets	Binning method	Spline method
ILT1	6.455 (6.063, 7.092)	4.750 (4.579, 4.955)
ILT2	0.752 (0.658, 0.903)	0.576 (0.538, 0.627)
ILT3	0.505 (0.465, 0.584)	0.428 (0.398, 0.463)

Source: Lee et al. [131]. With permission.

happen at the two ends of the average wind speed and standard deviation. This echoes the point made earlier that without binning the weather conditions, the spline method is able to make better use of the available data and overcome the problem of limited data for rare weather events.

Lee et al. [131] also note that the spline method, although conceptually and procedurally more involved, produces an overall model with fewer parameters. To see this, consider the following—for the three ILT datasets, the average $(K_\mu + K_\sigma)$ from the RJS algorithm is between 12 and 18. The number of model parameters d_k in Eq. 10.19 is generally less than 20, a number far smaller than the number of parameters used in the binning method. As explained in Section 10.3, when one uses a 6×10 grid to bin the two-dimensional wind covariates, the binning method in fact uses a total of 121 parameters for the overall model. Evidently, the spline method uses a sophisticated procedure to find a simpler model that is more capable.

10.6.4 Estimation of Extreme Load

Tables 10.5 and 10.6 show the estimates of the extreme load levels l_T, corresponding to $T = 20$ and $T = 50$ years, respectively. The values in parentheses are the 95% credible (or confidence) intervals.

One can observe that the extreme load levels, l_T, obtained by the binning

TABLE 10.6 Estimates of extreme load level ($l_T, T = 50$ years), unit: MN-m.

Datasets	Binning method	Spline method
ILT1	6.711 (6.240, 7.485)	4.800 (4.611, 5.019)
ILT2	0.786 (0.682, 0.957)	0.589 (0.547, 0.646)
ILT3	0.527 (0.480, 0.621)	0.438 (0.405, 0.476)

Source: Lee et al. [131]. With permission.

method are generally higher than those obtained by the spline method. This should not come as a surprise. As one pushes for a high quantile, more data would be needed in each weather bin but the amounts in reality are limited due to the binning method's compartmentalization of data. The binning method also produces a wider confidence interval than the spline method, as a result of the same rigidity in data handling. The procedure of computing the binning method's confidence interval is explained in Algorithm 10.1.

10.6.5 Simulation of Extreme Load

In this section, a simulation study is undertaken to assess the estimation accuracy of extreme load level in the long-term distribution. The simulations use one single covariate x, mimicking the wind speed, and a dependent variable z, corresponding to the maximum load. Algorithm 10.7 is used to generate the simulated data. A set of simulated data thus generated is included in the `Turbine Bending Moment Dataset` and ready to use, but interested readers are welcome to generate the simulated load response data by themselves.

Once the training dataset $\mathcal{D}_{\text{TR}}$ is simulated, both the binning method and spline method are used to estimate the extreme load levels l_T corresponding to two probabilities: 0.0001 and 0.00001. This estimation is based on drawing samples from the long-term distribution of z, as described in Section 10.4.4, which produces the posterior predictive distribution of l_T. To assess the estimation accuracy of the extreme quantile values, Lee et al. [131] also generate 100 additional simulated datasets, by repeating Step 1 through Step 3 in Algorithm 10.7, each of which consists of $100,000$ data points. For each dataset, one can compute the observed quantile values $l_{0.0001}$ and $l_{0.00001}$. Using the 100 simulated datasets, one can obtain 100 different samples of these quantiles.

Fig. 10.10(a) shows a scatter plot of the simulated x's and z's in $\mathcal{D}_{\text{TR}}$, which resembles the load responses observed. Figs. 10.10(b) and (c) present, under the two selected probabilities, the extreme load levels estimated by the two methods as well as the observed extreme quantile values. One notices that the binning method tends to overestimate the extreme quantile values and yields wider confidence intervals than the spline method. Furthermore, the degree of overestimation appears to increase as the probability corresponding

Algorithm 10.7 Simulated data generation to mimic wind speed and load response for assessing the long-term distribution.

1. Generate a sample x_i from a three-parameter Weibull distribution. Then sample x_{ij}, $j = 1, \ldots, 1,000$, from a normal distribution having x_i as its mean and a unit variance. The set of x_{ij}'s represents the different wind speeds within a bin.

2. Draw samples, z_{ij}, from a normal distribution with its mean as μ_{ij}^s and its standard deviation as σ_{ij}^s, which are expressed as follows:

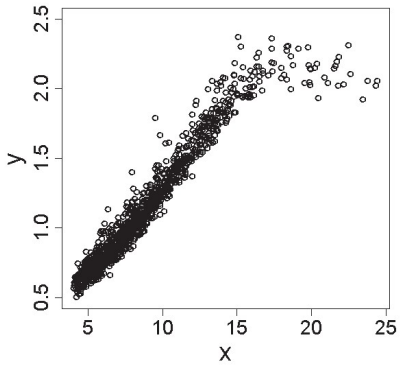
$$\mu_{ij}^s = \begin{cases} \frac{1.5}{[1+48\times\exp(-0.3\times x_{ij})]}, & \text{if} \quad x_i < 17, \\ \frac{1.5}{[1+48\times\exp(-0.3\times x_{ij})]} + [0.5 - 0.0016 \times (x_i + x_i^2)], & \text{if} \quad x_i \geq 17, \end{cases} \tag{10.35}$$

$$\sigma_{ij}^s = 0.1 \times \log(x_{ij}). \tag{10.36}$$

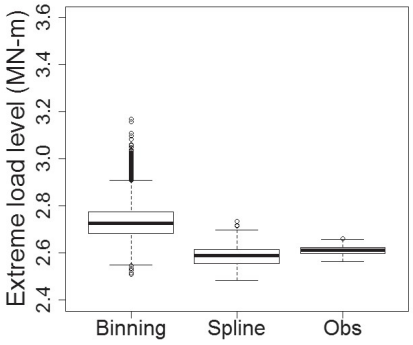
   The above set of equations is used to create a z response resembling the load data. The parameters used in the equations are chosen through trials so that the simulated z looks like the actual mechanical load response. While many of the parameters used above do not have any physical meaning, some of them do; for instance, the "17" in "$x_i < 17$" bears the meaning of the rated wind speed.

3. Find the maximum value $z_i = \max\{z_{i,1}, \ldots, z_{i,1000}\}$ and treat z_i as the maximum load response corresponding to x_i.

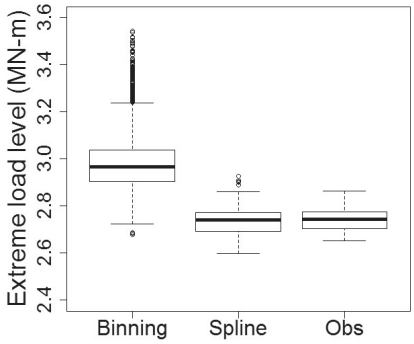
4. Repeat Step 1 through Step 3 for $i = 1, ..., 1,000$ to produce the training dataset with $n = 1,000$ data pairs, and denote this dataset by $\mathcal{D}_{\text{TR}} = \{(x_1, z_1), \ldots, (x_{1000}, z_{1000})\}$.



(a) Simulation dataset



(b) $l_{0.0001}$



(c) $l_{0.00001}$

FIGURE 10.10 Simulation dataset, estimated and observed extreme quantile values: (a) an example of the simulated dataset, (b) and (c) boxplots of the binning estimate, of the Bayesian spline estimate, and of the respective sample quantile across 100 simulated datasets. (Reprinted with permission from Lee et al. [131].)

to an extreme quantile value becomes smaller. This observation confirms what is observed in Section 10.6.4 using the field data.

GLOSSARY

BIC: Bayesian information criterion

cdf: Cumulative distribution function

G3: Three-parameter Gamma distribution

GEV: Generalized extreme value

GPD: Generalized Pareto distribution

GPL: Generalized piecewise linear (loss function)

IEC: International Electrotechnical Commission

IG3: Three-parameter inverse-Gaussian distribution

ILT: Inland turbine

LN3: Three-parameter log-normal distribution

MARS: Multivariate adaptive regression spline

MCMC: Markov chain Monte Carlo

MLE: Maximum likelihood estimation

pdf: Probability density function

PL: Piecewise linear (loss function)

POE: Probability of exceedance

POT: Peak over threshold

RAY: Rayleigh distribution

RJMCMC: Reserve jump Markov chain Monte Carlo

RJS: Reserve jump sampler

SIC: Schwarz information criterion

TN2: Two-parameter truncated normal distribution

UIP: Unit information prior

W2: Two-parameter Weibull distribution

W3: Three-parameter Weibull distribution

EXERCISES

10.1 In `R`, the package `evd` has a set of functions related to the reverse Weibull distribution. To generate values for the probability density function of reverse Weibull distribution, one can use the function `drweibull(x, loc, scale, shape, log = FALSE)`, where `loc`, `scale`, and `shape` are the three parameters to be specified. Their default values are 0, 1, and 1, respectively. Please use this function to plot a pdf curve for the reverse Weibull distribution by setting `loc` = 0. Compare the reverse Weibull pdf curve with the Weibull distribution pdf plot under the same values of their respective scale and shape parameters. For computing the Weibull pdf, please use the `dweibull` function in the `stats` package.

10.2 Plot the cdf and pdf curves for GEV distribution when, respectively, $\xi = 1$, $\xi = 0$, and $\xi = -1$. Make a note of the pattern of upper and lower tails under respective ξ values.

10.3 Understand the sensitivity of design load, l_T, to the parameters in a GEV distribution. Let $z \sim \text{GEV}(\mu, \sigma, \xi)$, where $\mu = 0$, $\sigma = 1$, and $\xi = 1$.

a. Compute l_T in Eq. 10.8 for $P_T = 3.8 \times 10^{-7}$.

b. Keeping $\sigma = 1$ and $\xi = 1$, change the location parameter such that the change in l_T is doubled (or halved). Quantify the change in the location parameter.

c. Keeping $\mu = 0$ and $\xi = 1$, change the scale parameter such that the change in l_T is doubled (or halved). Quantify the change in the scale parameter.

d. Keeping $\mu = 0$ and $\sigma = 1$, change the shape parameter such that the change in l_T is doubled (or halved). Quantify the change in the shape parameter.

e. Repeat the same exercise when the initial GEV distribution has the same μ and σ but $\xi = -1$.

f. Repeat the same exercise when the initial GEV distribution has the same μ and σ but $\xi = 0$.

10.4 To understand the binning method's lack of scalability, consider the following scenarios. Suppose that one has a full year of 10-minute data pairs, $\{\boldsymbol{x}_i, z_i\}$, with no missing data at all.

a. How many data pairs are there in this one-year dataset?

b. If $\boldsymbol{x}$ is one dimensional, and analysts use 10 bins for this variable, how many data points are there, on average, per bin?

c. What if $\boldsymbol{x}$ is three dimensional and each variable uses 10 bins, how many data points are there, on average, per bin? What if $\boldsymbol{x}$ is six dimensional?

d. Suppose that in order to adequately fit a GEV distribution with three constant parameters, one would need 25 data points. If we want to have sufficient amount of data points per bin to fit a GEV distribution for every single bin, what is the highest dimensionality of the input space that the binning method can serve?

10.5 Using ILT1 data in the `Turbine Bending Moment Dataset` and the binning method to estimate l_T corresponding to P_T of both 20-year service

and 50-year service. This time, instead of fixing the bin-based GEV distribution as the Gumbel distribution, please treat ξ as the third parameter in each bin and estimate based on the data. For the bins that do not have data or sufficient amount of data, follow the treatment in Algorithm 10.1. Compare the estimated $l_{20\text{-yr}}$ and $l_{50\text{-yr}}$ with its counterpart in Tables 10.5 and 10.6.

10.6 Suppose that we use a two-parameter Weibull distribution, instead of the three-parameter Weibull distribution, to model the average wind speed (recall that we used the two-parameter Weibull distribution to model the wind speed in Chapter 2). Reproduce the 95% credible interval plots in Fig. 10.6 and see if there is any noticeable difference.

10.7 Consider a TN2 model with constant η and δ, and, a LN2 model (two-parameter log-normal distribution) with constant distribution parameters, as alternatives for modeling wind speed standard deviation s. Use the SIC as the criterion to select the modeling option that produces the best model fit to the three turbines in the `Turbine Bending Moment Dataset`. Is the best model the TN2 with functional η and δ, TN2 with constant η and δ, or LN2 with constant parameters?

10.8 An alternative method to evaluate Eq. 10.2 is through a peak-over-threshold (POT) approach. The POT approach is to model the unconditional distribution of z directly without accounting for the effect of wind covariates in $\boldsymbol{x}$. First, select a threshold u for the z values in a dataset. Use the z data points above this threshold (that is where the name comes from) to estimate a generalized Pareto distribution (GPD). Assume that the extreme load z follows this GPD and estimate l_T for the corresponding P_T based on the estimated GPD. For the three turbine datasets, use this POT approach to estimate their 20-year and 50-year l_T and compare the outcome with those in Tables 10.5 and 10.6. When using the POT approach, set u as the 95-percentile value of the respective dataset.

10.9 Apply the POT method to the simulated training dataset, again with the threshold, u, set as the 95-percentile value of that dataset. Estimate $l_{0.0001}$ and $l_{0.00001}$, as what has been done while using the binning and spline methods. Compare the POT outcome with those in Fig. 10.10.

CHAPTER 11

Computer Simulator-Based Load Analysis

The principal challenge in reliability assessment of wind turbines is rooted in the fact that a small tail probability, $f(z > l_T)$, in the order of 10^{-7} for $T = 50$ years, needs to be estimated. To accurately estimate this type of small probability requires a sufficient number of high-z load response values. If one opts to collect enough high-z values from physical turbine systems, it takes tens of years, as the high-z values, by definition, are rare events. Adding to the challenge is that hardly have any commercial wind turbines been installed with strain sensors, due to cost concerns. Physically measured bending moments are typically obtained on a few test turbines and only for a short duration of time, which is the reason behind the need for an extrapolation and the modeling of the conditional load density, as explained in Chapter 10. Wind engineers have been developing aeroelastic simulators that can produce reasonably trustworthy bending moments response under a wind force input. The availability of these simulators lends a degree of convenience to load analysis, as a simulator can be steered, at least in principle, towards the region of high load responses so as to produce more high-z data points. For this reason, using the aeroelastic simulators could expedite and enhance the estimation of extreme load distribution and facilitate the reliability assessment of wind turbines. Of course, running aeroelastic turbine load simulators can be computationally expensive. Data science methods are much needed to make the simulator-based load analysis efficient and practical.

11.1 TURBINE LOAD COMPUTER SIMULATION

11.1.1 NREL Simulators

Two NREL simulators are popularly used in the study to generate the structural load response on a turbine—one is TurbSim [112] and the other is FAST [113]. To simulate the structural load response on a turbine, it takes two steps: first, TurbSim generates an inflow wind profile in front of a wind turbine, and second, FAST takes the inflow profile as input and simulates structural and mechanical load responses at multiple turbine components. Please refer to Fig. 10.1 for an illustration of load responses on turbine components.

More than a single-point turbulence computed based on the hub height wind speed, TurbSim simulates a full-field stochastic inflow turbulence environment in front of a turbine, "*reflect[ing] the proper spatiotemporal turbulent velocity field relationships seen in instabilities associated with nocturnal boundary layer flows*" [120]. The input to TurbSim is the hub height wind speed, either measured or simulated, and the output is the full-field inflow environments to be used to drive a downstream load simulator. FAST is the aeroelastic dynamic load simulator, and uses wind inflow data and solves for the rotor wake effects and blade-element loads.

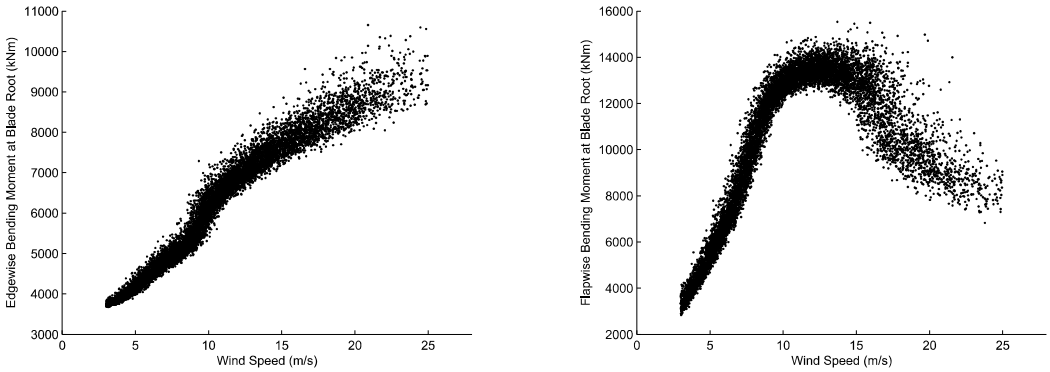
The data in the `Simulated Bending Moment Dataset` are simulated using the two simulators [37]. According to Choe et al. [37], the 10-minute average wind speed is simulated using a Rayleigh distribution. Recall that as mentioned in Section 10.4.3, the IEC standard recommends using a Rayleigh distribution to model the 10-minute average wind speed, although the numerical studies in Section 10.6.1 show that other distributions may fit the actual wind speed data better. TurbSim and FAST are used to simulate a turbine's 10-minute operations, given the average wind speed. The maximum load responses at a blade root are recorded as the output. Two load types, the edgewise and flapwise bending moments, are simulated and their respective maximum values in a 10-minute interval are recorded in the `Simulated Bending Moment Dataset`. Choe et al. [37] define a simulation replication or one simulator run as a single execution of the 10-minute turbine operation simulation that generates a 10-minute maximum edgewise and flapwise load. The *simulated* maximum load is still denoted by z, same as the notation used for the physical maximum load.

Fig. 11.1 illustrates the load responses simulated from TurbSim and FAST, following the procedure discussed in [149].

11.1.2 Deterministic and Stochastic Simulators

Not only does the wind industry use computer simulators to complement physical experiments or make up data deficiency in physical measurements, the use of computer simulators, sometimes referred to as *computer experiments*, is popular and common in many other engineering applications [124, 193].

Many computer simulators used in engineering applications are based on



(a) Edgewise bending moments (b) Flapwise bending moments

FIGURE 11.1 Simulated blade root load responses. (Reprinted with permission from Choe et al. [37].)

solving a set of partial differential equations, or a mix of differential and algebraic equations, derived from physical laws and principles. Finite element analysis in mechanics is a frequently mentioned example of this type of computer simulators. These computer simulators are referred to as *deterministic* computer simulators, and numerical analyses run on such simulators are called deterministic computer experiments. They are called "deterministic" because for the same given input, the simulator's output remains the same, no matter how many times one re-runs the same simulator. Let us denote by $g(\cdot)$ the function of the black-box simulator, and by $z(\boldsymbol{x}) := g(\boldsymbol{x})$ the output of the simulator, given the input at $\boldsymbol{x}$. Note that z here refers to a generic output, although it could be, but not necessarily, the load response. For a deterministic computer simulator, $z(\boldsymbol{x})$ does not change, as long as $\boldsymbol{x}$ stays the same.

The development of turbine load simulators is indeed based on aerodynamic and aeroelastic physical principles. But TurbSim and FAST are not deterministic simulators, because for a given input, $\boldsymbol{x}$, the load response is not guaranteed to be the same. Rather, the simulator response exhibits randomness, resembling the characteristics of noisy physical measurements. This is because the turbine load simulators embed a large number of uncontrollable variables inside the black-box simulator. These variables take different values, produced from certain random number generators, at individual runs of the same simulator, so that even if the input, $\boldsymbol{x}$, remains the same, the output, z, could be, and is almost surely different. The turbine load simulators are therefore known as the *stochastic* computer simulators, and numerical analyses run on them are stochastic simulations or stochastic computer experiments, mimicking physical experiments. For the stochastic simulators, their g function should include two types of inputs—$\boldsymbol{x}$ that can be set prior to running the simulator and $\boldsymbol{\epsilon}$ that is not controlled explicitly but takes its values from ran-

dom number generators. In other words, the simulator response z is such that $z(\boldsymbol{x}) = g(\boldsymbol{x}, \boldsymbol{\epsilon})$.

When simulating the turbine load response, $\boldsymbol{x}$ is considered a stochastic input, and its marginal distribution, $f(\boldsymbol{x})$, is either known *a priori*, or practically, estimated from the historical data. Random samples of $\boldsymbol{x}$ are drawn from $f(\boldsymbol{x})$ and used to drive TurbSim and FAST simulators. For a deterministic simulator, its inputs can be stochastic and drawn from its own marginal distribution. Analysts may question whether a stochastic simulator may become deterministic, if one treats the combined set of variables, $(\boldsymbol{x}, \boldsymbol{\epsilon})$, as a new input. This is to say, let us specify the joint probability distribution, $f(\boldsymbol{x}, \boldsymbol{\epsilon})$, for both $\boldsymbol{x}$ and $\boldsymbol{\epsilon}$ and then draw samples from $f(\boldsymbol{x}, \boldsymbol{\epsilon})$ to drive the stochastic simulator. Given a specific value of $(\boldsymbol{x}, \boldsymbol{\epsilon})$, the simulator response, $g(\boldsymbol{x}, \boldsymbol{\epsilon})$, remains the same, no matter how many times the same value of $(\boldsymbol{x}, \boldsymbol{\epsilon})$ is used to re-run the same simulator.

Technically, this is correct. In fact, nothing is exactly uncontrolled in computer simulations—even the random numbers generated are, rigorously speaking, pseudo random numbers. But practically, there are too many random variables embedded inside the load response simulators to be specified with a joint probability distribution. According to Choe et al. [37], $\boldsymbol{\epsilon}$ in the NREL simulators has over eight million elements. By contrast, elements in $\boldsymbol{x}$ are far fewer—its number is generally in a single digit. It is thus practical to specify a joint distribution only for $\boldsymbol{x}$ and draw samples from it, while leaving $\boldsymbol{\epsilon}$ to be individually handled by its own random number generator. A computer simulator is stochastic in the sense that $\boldsymbol{\epsilon}$ is left uncontrolled.

Another branch of stochastic simulators are commonly found in discrete event simulations. One of such simulations is mentioned in Section 9.4.3, which is the DEVS-based simulation platform for a wind farm. In that wind farm simulator, a number of inputs or parameters, such as the number and locations of wind turbines, can be specified by analysts running the simulation, but there are many more random variables left to be individually handled by a respective random number generator, such as the degradation path for a turbine component. In the end, even under a fixed $\boldsymbol{x}$, the wind farm simulator changes its response when it is re-run.

11.1.3 Simulator versus Emulator

Running computer simulators is to reduce cost by not conducting too many physical experiments, either too expensive, or too time consuming, or unrealistic. But running computer simulators incurs its own cost, in the form of computational expense. Depending on the fidelity of a computer simulator, the time to run one simulation replication ranges from a couple of minutes (low-fidelity ones) to hours or even days (high-fidelity ones).

Analysts therefore develop efficient, or computationally cheap, mathematical surrogates of computer simulators and hope to rein in the computational expense by running a small number of computer simulations but a large num-

ber of the surrogate models. The surrogate models are models of models, because computer simulators are themselves mathematical models of a physical reality, rather than the physical reality itself. For this reason, a surrogate model is known as a meta-model. They are also called *emulators*, to be differentiated from the simulators, and the surrogate models do mean to emulate the behavior of a respective simulator.

A popular branch of emulators is based on Gaussian process regression, or the kriging model, as introduced in Section 3.1.3. To model simulator responses, the location input, $\mathbf{s}$, used in the spatial modeling in Section 3.1.3, is to be replaced by a generic input, $\boldsymbol{x}$. While Gaussian process regression used in Section 3.1.3 for spatial modeling has an input dimension of two, the same modeling approach can be easily extended to general applications of more than two inputs, without changing much of the formulations and solution procedures as outlined in Section 3.1.3.

When modeling a stochastic simulator response, Eq. 3.8 or Eq. 3.17 can be directly used, as the simulator response is treated as if it were a physical response. A training dataset, collected from running the stochastic simulators at different $\boldsymbol{x}$'s, is needed to estimate the parameters in the Gaussian process model. The resulting model, if we express it by $\hat{g}(\boldsymbol{x})$, is a meta-model or an emulator.

When modeling a deterministic simulator response, the main difference is to use Eq. 3.8 or Eq. 3.17 without the nugget effect, i.e., remove ε in the respective equation. This is because a deterministic simulator returns the same response for the same input, so that an emulator is supposed to produce the precise response at the same input value of $\boldsymbol{x}$. It can be shown that a Gaussian regression model without the nugget effect interpolates precisely through the training data points, known as its interpolating property (see Exercise 3.3).

The popularity of the Gaussian process model as an emulator arises from its modeling of deterministic computer simulators. When deterministic computer simulators become common, analysts realize its difference from physical experiments, particularly the aspect of having noise-free responses, and therefore seek a different modeling approach. Sacks and his co-authors adopt the Gaussian process model from spatial statistics to model computer experiments [189, 190] and note the interpolating property; their effort launched the field of design and analysis of computer experiments.

But Gaussian process models are not the only emulator choice, especially when it comes to modeling the stochastic computer simulators. Recall that the response of a stochastic computer simulator looks more like physical measurements. Many data science methods introduced in this book, employed to model various types of physical responses, can be used to model the response of a stochastic computer simulator and hence be an emulator. As we will see in Section 11.4, the emulator used in the turbine load analysis is not a Gaussian process model.

11.2 IMPORTANCE SAMPLING

Let us first consider the use of deterministic computer simulator in reliability analysis. Given a wind condition $\boldsymbol{x}$, the deterministic computer simulator produces a load response, $z = g(\boldsymbol{x})$. This output can be compared with the design load, or a turbine system's resistance level, l, to see if the turbine structure may fail under the simulated load response z. For reliability assessment, analysts are interested in knowing the failure probability $P(z > l)$, which was expressed in Eq. 10.1 with a subscript T. Here we drop the subscript for the simplicity of notation. Relying on the response of a deterministic computer simulator, this failure probability can be expressed as

$$P(z > l) = \int \mathbb{1}(g(\boldsymbol{x}) > l) f(\boldsymbol{x}) d\boldsymbol{x} = \mathbb{E}[\mathbb{1}(g(\boldsymbol{x}) > l)], \tag{11.1}$$

where $\mathbb{1}(\cdot)$ is the indicator function.

11.2.1 Random Sampling for Reliability Analysis

Computer simulators, including the turbine load simulators, are considered black boxes because an output is numerically computed by going through thousands of lines of computer codes. It is impractical to analytically evaluate the failure probability $P(z > l)$ in Eq. 11.1. It is, however, rather straightforward to evaluate the failure probability empirically through random sampling. The simplest method is the plain version Monte Carlo method, also known as the crude Monte Carlo (CMC), which is to draw random samples, $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_T}\}$, from $f(\boldsymbol{x})$, where N_T is the number of the random samples. Each one of the samples is used to drive the computer simulator and produce a corresponding load output. As such, N_T is also the number of simulation runs.

The simulated load response is then compared with l. If $g(\boldsymbol{x}) > l$, a failure occurs and the indicator function, $\mathbb{1}(g(\boldsymbol{x}_i) > l)$, returns a one; otherwise, no failure occurs and the indicator function, $\mathbb{1}(g(\boldsymbol{x}_i) > l)$, returns a zero. The failure probability is empirically estimated by

$$\hat{P}(z > l) = \frac{1}{N_T} \sum_{i=1}^{N_T} \mathbb{1}(g(\boldsymbol{x}_i) > l). \tag{11.2}$$

The estimate is simply counting how many times, among the N_T runs, the simulator output exceeds the design load level l.

The crude Monte Carlo method is easy to use and applies to almost any applications. Its main shortcoming is the inefficiency for reliability assessment. Heidelberger [89] presents the following example to stress the point. Let us denote the probability in Eq. 11.1 by P and the estimate in Eq. 11.2 by $\hat{P}_{\text{CMC}}$. It is not difficult to show (see Exercise 11.1) that the expectation and variance

of $\hat{P}_{\text{CMC}}$ are, respectively,

$$\begin{aligned} \mathbb{E}[\hat{P}_{\text{CMC}}] &= P, \quad \text{and} \\ Var[\hat{P}_{\text{CMC}}] &= \frac{1}{N_T} P(1-P). \end{aligned} \tag{11.3}$$

If using a normal approximation, the $100(1-\alpha)\%$ confidence interval for P is $\hat{P}_{\text{CMC}} \pm z_{\alpha/2}\sqrt{P(1-P)/N_T}$. A similar treatment is used in Eq. 2.8. The expectation expression in Eq. 11.3 also means that the crude Monte Carlo estimate is unbiased.

Heidelberger [89] asks that how many random samples, or equivalently, how many simulator runs, are required in order to estimate the 99% confidence interval of P to be within 10% of the true probability. To accomplish the desired estimation accuracy, it requires that $z_{\alpha/2}\sqrt{P(1-P)/N_T} \leq 0.1P$ for $\alpha = 0.01$, or equivalently,

$$2.58\sqrt{\frac{(1-P)}{P} \cdot \frac{1}{N_T}} \leq 0.1,$$

so that

$$N_T \geq 666 \times \frac{1-P}{P}. \tag{11.4}$$

For a well-designed product, its failure probability P is small, suggesting $1 - P \approx 1$, so that N_T is roughly of $666/P$, which is going to be large for a small P. Suppose the target failure probability is at the level of $P = 10^{-5}$. To have an accurate enough estimate of this small probability, the sample size or the number of simulation runs required is 6.7×10^7. Even if a single run of the computer simulator takes only one second, 6.7×10^7 seconds still translate to more than two years. The essence of reliability assessment is to capture and characterize the behavior of rare events. While attempting to come up with enough samples of the rare events, the inefficiency of the crude Monte Carlo leads to a high computational demand.

11.2.2 Importance Sampling Using Deterministic Simulator

Importance sampling is to introduce another density function, $q(\boldsymbol{x})$, to draw samples of $\boldsymbol{x}$, where $q(\boldsymbol{x})$ is referred to as the importance sampling density. We explain later where the name comes from.

While using $q(\boldsymbol{x})$, the failure probability expression in Eq. 11.1 can be written differently, i.e.,

$$P(z > l) = \int \mathbb{1}(g(\boldsymbol{x}) > l)\frac{f(\boldsymbol{x})}{q(\boldsymbol{x})} q(\boldsymbol{x}) d\boldsymbol{x} = \mathbb{E}\left[\mathbb{1}(g(\boldsymbol{x}) > l)\frac{f(\boldsymbol{x})}{q(\boldsymbol{x})}\right]. \tag{11.5}$$

By multiplying and dividing $q(\boldsymbol{x})$ in the integrand, the above probability expression remains equivalent to that in Eq. 11.1.

Denote by

$$\mathcal{L}(\boldsymbol{x}) = \frac{f(\boldsymbol{x})}{q(\boldsymbol{x})}$$

the likelihood ratio between the two density functions. Eq. 11.5 can be expressed as

$$P(z > l) = \mathbb{E}[\mathbb{1}(g(\boldsymbol{x}) > l)\mathcal{L}(\boldsymbol{x})].$$

The empirically estimated failure probability based on importance sampling (IS) density is then

$$\hat{P}_{\text{IS}}(z > l) = \frac{1}{N_T}\sum_{i=1}^{N_T} \mathbb{1}(g(\boldsymbol{x}_i) > l)\mathcal{L}(\boldsymbol{x}_i), \tag{11.6}$$

where the samples, $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_T}\}$, are drawn from $q(\boldsymbol{x})$.

Technically, any valid density function can be used as $q(\boldsymbol{x})$ in importance sampling, and $\hat{P}_{\text{IS}}$ is an unbiased estimator of P, as long as $q(\boldsymbol{x}) = 0$ implies that $\mathbb{1}(g(\boldsymbol{x}) > l)f(\boldsymbol{x}) = 0$ for any $\boldsymbol{x}$, which means that a non-zero feasible sample under the old density $f(\cdot)$ with $g(\boldsymbol{x}) > l$ must also be a non-zero feasible sample under the new density $q(\cdot)$. However, this does not mean that an arbitrary choice of $q(\boldsymbol{x})$ can help address the computational inefficiency problem of the crude Monte Carlo method. To understand the choice for an optimal importance sampling density, we first provide an intuitive understanding how importance sampling works.

The condition to be verified for failures, $g(\boldsymbol{x}) > l$, defines the events of interest (EOI) for a reliability assessment. But the concentration of $f(\boldsymbol{x})$ does not coincide with the EOI. The region of $\boldsymbol{x}$, whose corresponding response belongs to the EOI, is referred to as the *critical region*. By the nature that the EOI in reliability analysis are rare, random sampling from $f(\cdot)$ has a low hit rate on the critical region. An importance sampling can help if the density so chosen, $q(\cdot)$, steers the sample concentration towards the critical region. This means that while $f(\cdot)$ is small over the critical region, $q(\cdot)$ needs to be large on that region, so as to make the EOI likely to occur. The name, "importance," is given to the sampling approach because the new density is supposed to place the right importance on the critical region, or the new density concentrates on the region of importance.

This intuition is realized through variance reduction in random sampling. To see this, consider the following. The variance of the importance sampling estimator in Eq. 11.6 can be expressed as

$$\begin{aligned} Var[\hat{P}_{\text{IS}}] &= \frac{1}{N_T^2}\sum_{i=1}^{N_T} \mathbb{E}_q\left[(\mathbb{1}(g(\boldsymbol{x}_i) > l)\mathcal{L}(\boldsymbol{x}_i))^2\right] + C \\ &= \frac{1}{N_T}\mathbb{E}_f\left[\mathbb{1}(g(\boldsymbol{x}) > l)\mathcal{L}(\boldsymbol{x})\right] + C, \end{aligned} \tag{11.7}$$

where the subscript placed on the expectation operator is to make explicit

which probability measure the expectation is taken with respect to and C is a constant not depending on the sampling action. The first equality in the above question means that reducing the variance of the importance sampling estimator corresponds to selecting a $q(\boldsymbol{x})$ that reduces the second moment of $\mathbb{1}(g(\boldsymbol{x}) > l)\mathcal{L}(\boldsymbol{x})$.

Let us take a look at the likelihood ratio, which is $\mathcal{L}(\boldsymbol{x}) = f(\boldsymbol{x})/q(\boldsymbol{x})$. For importance sampling, following the intuition above, $f(\boldsymbol{x})$ is small in the critical region where $\mathbb{1}(g(\boldsymbol{x}_i) > l) = 1$, while $q(\boldsymbol{x})$ should be large. As such, the likelihood ratio, $\mathcal{L}(\boldsymbol{x})$, is small. Consequently, the variance of the importance sampling is small, according to Eq. 11.7. A proper choice of the importance sampling density is thereby to reduce the likelihood ratio, which in turn makes the samples less spread out (small variance). Together with the unbiasedness property of $\hat{P}_{\text{IS}}$, a variance-reduced importance sampling is able to concentrate on the critical region to sample. For the derivation of Eq. 11.7, please see Exercise 11.2.

The theoretically optimal importance sampling density is

$$q^*_{\text{IS}} = \frac{\mathbb{1}(g(\boldsymbol{x}) > l)f(\boldsymbol{x})}{P(z > l)}, \tag{11.8}$$

because this q^*_{IS} leads to a failure probability estimate that has a zero (and hence the smallest) variance, and one sample from it gives us the unconditional POE, $P(z > l)$, exactly. Practically, this q^*_{IS} is not implementable. The probability $P(z > l)$ is unknown and precisely what analysts want to estimate using the simulators and random samples. Moreover, the critical region, implied by $g(\boldsymbol{x}) > l$, is not known, either, before the simulator is run on the random samples of $\boldsymbol{x}$.

De Boer et al. [45] present a cross-entropy-based approximation to implement the idea of importance sampling. Consider the case that the density function, $f(\boldsymbol{x})$, can be parameterized by a vector $\boldsymbol{u}$. To make this parametrization explicit, let us express it as $f(\boldsymbol{x}; \boldsymbol{u})$. Suppose that the importance sampling density takes the same function form but uses different parameters, i.e., $q(\boldsymbol{x}) := f(\boldsymbol{x}; \boldsymbol{v})$. The likelihood ratio can be expressed as

$$\mathcal{L}(\boldsymbol{x}; \boldsymbol{u}, \boldsymbol{v}) = \frac{f(\boldsymbol{x})}{q(\boldsymbol{x})} = \frac{f(\boldsymbol{x}; \boldsymbol{u})}{f(\boldsymbol{x}; \boldsymbol{v})}. \tag{11.9}$$

The cross-entropy algorithm is iterative in nature. When the algorithm starts, it attempts to find an event not so rare, by setting a probability, say $\kappa = 0.01$, so that there are almost surely EOI produced from the simulator. Let t be the iteration index and N_t be the sample size at the t-th iteration. When the N_t samples are evaluated using the simulator at the t-th iteration, the responses are labeled as $\{g_1^{(t)}, \ldots, g_{N_t}^{(t)}\}$. Without ambiguity, the superscript (t) is often dropped. We order the simulator response from smallest to largest, such that $g_{(1)} \leq g_{(2)} \leq \ldots \leq g_{(N_t)}$, where $g_{(j)}$ is the j-th order-statistic of the sequence $\{g(\boldsymbol{x}_1), \ldots, g(\boldsymbol{x}_{N_t})\}$.

The iterative cross-entropy algorithm constructs a sequence of reference parameters $\{\boldsymbol{v}_t, t \geq 0\}$ and design load thresholds $\{l_t, t \geq 1\}$. It starts with $\boldsymbol{v}_0 = \boldsymbol{u}$ and updates both $\boldsymbol{v}_t$ and l_t by steering the sampling action towards the critical region. The specific steps are outlined in Algorithm 11.1. The optimization formulation in Step 3 is based on the minimization of the Kullback-Leibler distance between the optimal importance sampling density, q^*_{IS} in Eq. 11.8, and the actual importance sampling density to be used for the next iteration, $q^{(t+1)}(\boldsymbol{x}) := f(\boldsymbol{x}; \boldsymbol{v}_t)$. The Kullback-Leibler distance is also termed the *cross entropy* between the two density functions of interest (see Exercise 11.3).

Algorithm 11.1 Iterative cross-entropy approximation for importance sampling.

1. Set $\hat{\boldsymbol{v}}_0 = \boldsymbol{u}$ and $t = 1$.

2. Draw samples, $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_t}\}$, from the density $q^{(t)}(\boldsymbol{x}) := f(\boldsymbol{x}; \boldsymbol{v}_{t-1})$. Compute the $(1-\kappa)N_t$-th order-statistic of $\{g(\boldsymbol{x}_1), \ldots, g(\boldsymbol{x}_{N_t})\}$ and set that as the estimate of l_t, i.e.,

$$\hat{l}_t = g_{(\lceil (1-\kappa)N_t \rceil)}.$$

   If $\hat{l}_t \geq l$, then let $\hat{l}_t = l$.

3. Use the same samples drawn in Step 2, $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N_t}\}$, to solve the following optimization problem and get an update of $\boldsymbol{v}_t$. Denote the solution by $\hat{\boldsymbol{v}}_t$.

$$\max_{\boldsymbol{v}} \frac{1}{N_t} \sum_{i=1}^{N_t} \mathbb{1}(g(\boldsymbol{x}_i \geq \hat{l}_t)) \mathcal{L}(\boldsymbol{x}_i; \boldsymbol{u}, \hat{\boldsymbol{v}}_{t-1}) \ln f(\boldsymbol{x}_i; \boldsymbol{v}). \tag{11.10}$$

4. If $\hat{l}_t < l$, set $t = t + 1$ and reiterate from Step 2. Else proceed to Step 5.

5. Estimate the failure probability by using Eq. 11.6, re-written below as

$$\hat{P}_{\text{IS}}(z > l) = \frac{1}{N_T} \sum_{i=1}^{N_T} \mathbb{1}(g(\boldsymbol{x}_i) > l) \mathcal{L}(\boldsymbol{x}_i; \boldsymbol{u}, \hat{\boldsymbol{v}}_T).$$

   where T is the final number of iterations.

Dubourg et al. [53] present a different approximation approach, which is based on the use of a meta-model. The idea is simple. First, draw a small number of samples of $\boldsymbol{x}$, say a couple of hundreds, and use the computer simulator to generate the corresponding structural responses. Using this small set of simulator-generated samples, Dubourg et al. [53] build a Gaussian process emulator, which can run more efficiently and be used to generate a much larger

sample set, say several thousands or even tens of thousands. The importance sampling estimate in Eq. 11.6, instead of relying on the simulator function $g(\cdot)$, now uses the emulator function, $\hat{g}(\cdot)$.

One challenge faced by this meta-model-based approach is that with the initial small number of samples, the chance of having a sufficient number of EOI is low. The subsequent Gaussian process emulator is therefore unlikely able to gain a good accuracy in the tail probability estimation when there are very few quality samples to build the meta-model in the first place. Like the cross-entropy approach, an iterative procedure appears unavoidable for the meta-model-based approach, which gradually steers the sampling action towards the critical region.

11.3 IMPORTANCE SAMPLING USING STOCHASTIC SIMULATORS

The importance sampling described in Section 11.2 relies on the use of a deterministic computer simulator. This is reflected in the failure verification function, $\mathbb{1}(g(\boldsymbol{x}) > l)$. Due to the deterministic nature of the simulator used, $g(\boldsymbol{x})$ is a constant for a given $\boldsymbol{x}$, so that $g(\boldsymbol{x}) > l$ is either true or false, meaning $\mathbb{1}(g(\boldsymbol{x}) > l)$ is either one or zero, once $\boldsymbol{x}$ is fixed. This is no longer true for stochastic simulators, because $g(\boldsymbol{x})$ varies even for the same $\boldsymbol{x}$. The verification condition, $g(\boldsymbol{x}) > l$, compares in fact a random variable with a threshold, and for this reason, the indicator function is no longer appropriate to be used to capture the failure verification outcome. Rather, a probability should be assessed of this condition, namely $P(g(\boldsymbol{x}) > l)$.

In the context of stochastic simulators, the crude Monte Carlo estimate is changed to

$$\begin{aligned} \hat{P}_{\mathrm{CMC}}(z > l) &= \frac{1}{M} \sum_{i=1}^{M} \hat{P}(g(\boldsymbol{x}_i) > l) \\ &= \frac{1}{M} \sum_{i=1}^{M} \left(\frac{1}{N_i} \sum_{j=1}^{N_i} \mathbb{1}(g_j(\boldsymbol{x}_i) > l) \right), \end{aligned} \tag{11.11}$$

where $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M\}$ are M random samples from $f(\cdot)$ and M is called the *input sample size*. At each input $\boldsymbol{x}_i$, the simulator is run N_i times to produce N_i outputs, $g_1(\boldsymbol{x}_i), \ldots, g_{N_i}(\boldsymbol{x}_i)$, each of which is a realization of a stochastic process and can then be compared with the design threshold l in a deterministic manner. The number of simulations per input, N_i, is called the *allocation size*. The total number of simulator runs is then $N_T = \sum_{i=1}^{M} N_i$.

Apparently, the inclusion of the inner summation in Eq. 11.11 is the major difference between the failure probability estimate using a stochastic simulator and that using a deterministic simulator. When using a deterministic simulator, N_i is set simply one, so that $N_T = M$. When using a stochastic simulator,

the sample average of N_i simulator responses under the same $\boldsymbol{x}_i$ is used to approximate the probability, $\hat{P}(g(\boldsymbol{x}_i) > l)$.

Importance sampling based on deterministic simulators can be explicitly referred to as the deterministic importance sampling (DIS), whereas importance sampling based on stochastic simulators is referred to as the stochastic importance sampling (SIS). In the sequel, some of the "IS" subscripts used previously is replaced by "DIS." For instance, q^*_{IS} in Eq. 11.8 is expressed as q^*_{DIS} from this point onwards.

Choe et al. [37] develop two versions of the stochastic importance sampling method, referred to as SIS1 and SIS2, respectively, which are to be explained in the sequel.

11.3.1 Stochastic Importance Sampling Method 1

Noticing the difference between Eq. 11.2 and Eq. 11.11, when introducing an importance sampling density to the stochastic simulators, Eq. 11.6 should be written as

$$\hat{P}_{\text{SIS1}}(z > l) = \frac{1}{M}\sum_{i=1}^{M} \hat{P}(g(\boldsymbol{x}_i) > l)\mathcal{L}(\boldsymbol{x}_i) = \frac{1}{M}\sum_{i=1}^{M}\left(\frac{1}{N_i}\sum_{j=1}^{N_i}\mathbb{1}(g_j(\boldsymbol{x}_i) > l)\right)\mathcal{L}(\boldsymbol{x}_i), \tag{11.12}$$

where the samples, $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M\}$, are drawn from $q(\boldsymbol{x})$. Here, $P(g(\boldsymbol{x}_i) > l)$ is the probability of exceedance, conditioned on input $\boldsymbol{x}_i$. Let us denote this conditional POE by

$$S(\boldsymbol{x}) := P(g(\boldsymbol{x}) > l). \tag{11.13}$$

In Eq. 11.12, the conditional POE is estimated by the sample mean of successes.

In SIS1, Choe et al. [37] state that N_T and M are assumed given and the goal is to find the optimal allocation, N_i, and the optimal importance sampling density function, $q_{\text{SIS1}}(\cdot)$.

Recall the intuition behind importance sampling described in Section 11.2.2. The optimal importance sampling density is supposed to minimize the variance of the failure probability estimate. For $\hat{P}_{\text{SIS1}}(z > l)$, Choe et al. [37] obtain

$$\begin{aligned} Var[\hat{P}_{\text{SIS1}}] &= Var\left[\frac{1}{M}\sum_{i=1}^{M}\hat{S}(\boldsymbol{x}_i)\mathcal{L}(\boldsymbol{x}_i)\right] \\ &= \frac{1}{M^2}\mathbb{E}\left[Var\left\{\sum_{i=1}^{M}\hat{S}(\boldsymbol{x}_i)\mathcal{L}(\boldsymbol{x}_i)\right\}\right] + \frac{1}{M^2}Var\left[\mathbb{E}\left\{\sum_{i=1}^{M}\hat{S}(\boldsymbol{x}_i)\mathcal{L}(\boldsymbol{x}_i)\right\}\right] \\ &= \frac{1}{M^2}\mathbb{E}\left[\sum_{i=1}^{M}\frac{1}{N_i}S(\boldsymbol{x}_i)(1 - S(\boldsymbol{x}_i))(\mathcal{L}(\boldsymbol{x}_i))^2\right] + \frac{1}{M}Var\left[S(\boldsymbol{x})\mathcal{L}(\boldsymbol{x})\right]. \end{aligned} \tag{11.14}$$

Choe et al. further prove that the following allocation sizes and importance

sampling density function make $\hat{P}_{\text{SIS1}}$ an unbiased estimator and minimize the variance of the failure probability estimate in Eq. 11.14:

$$q^*_{\text{SIS1}}(\boldsymbol{x}) = \frac{1}{C_{q1}} f(\boldsymbol{x}) \sqrt{\frac{1}{N_T} S(\boldsymbol{x})(1 - S(\boldsymbol{x})) + S(\boldsymbol{x})^2}, \tag{11.15a}$$

$$N_i^* = \frac{N_T \sqrt{\frac{N_T(1-S(\boldsymbol{x}_i))}{1+(N_T-1)S(\boldsymbol{x}_i)}}}{\sum_{j=1}^{M} \sqrt{\frac{N_T(1-S(\boldsymbol{x}_j))}{1+(N_T-1)S(\boldsymbol{x}_j)}}}, \tag{11.15b}$$

where C_{q1} is a normalizing constant such that

$$C_{q1} = \int f(\boldsymbol{x}) \sqrt{\frac{1}{N_T} S(\boldsymbol{x})(1 - S(\boldsymbol{x})) + S(\boldsymbol{x})^2}\, d\boldsymbol{x}.$$

When using the above formula for N_i, N_i is rounded to the nearest integer. If the rounding yields a zero, Choe et al. suggest using one in its place in order to ensure unbiasedness in the failure probability estimation.

The importance sampling density is a re-weighted version of the original density for $\boldsymbol{x}$. It gives more weight to the critical region when EOI are more likely to occur, and less weight to the region when EOI do not happen as often, so as to refocus the sampling effort on the critical region.

The allocation size is roughly proportional to $\sqrt{1 - S(\boldsymbol{x}_i)}$, after we approximate $1 + (N_T - 1)S(\boldsymbol{x}_i)$ by one for a small $S(\boldsymbol{x}_i)$. This allocation policy says that for a smaller failure probability, one needs a larger size of samples. This result may sound counter-intuitive at first, because one would expect the smaller failure probability area to be accompanied by a smaller sample size. While sampling from the critical region where EOI are more likely to occur, the optimal importance sampling density, q^*_{SIS1}, concentrates more resources on the region where $g(\cdot)$ is close to l, rather than on the region where $g(\cdot)$ is much greater than l. This turns out to be a good strategy because for the region where $g(\cdot)$ is much greater than l, the certainty is high, foreclosing the need for large sample sizes. In summary, among the important input conditions under which a system can possibly fail, SIS1's allocation strategy finds it a more judicious use of the simulation resources by allocating a larger (smaller) number of replications in the region with a relatively small (large) $S(\boldsymbol{x})$.

The q^*_{SIS1} reduces to q^*_{DIS} in Eq. 11.8 (where it was called q^*_{IS} then) when the stochastic simulator is replaced by a deterministic simulator. Under a deterministic simulator, $N_i = 1$, $N_T = M$, and $S(\boldsymbol{x}) = \mathbb{1}(g(\boldsymbol{x}) > l)$. The last expression means that under a deterministic simulator, the conditional POE deteriorates to an indicator function, taking either zero or one. As such, $S(\boldsymbol{x})(1 - S(\boldsymbol{x})) = 0$, so that the density function becomes

$$q^*_{\text{SIS1}} = \frac{S(\boldsymbol{x})f(\boldsymbol{x})}{\int S(\boldsymbol{x})f(\boldsymbol{x})d\boldsymbol{x}} = \frac{\mathbb{1}(g(\boldsymbol{x}) > l)f(\boldsymbol{x})}{\int \mathbb{1}(g(\boldsymbol{x}) > l)f(\boldsymbol{x})d\boldsymbol{x}} = \frac{\mathbb{1}(g(\boldsymbol{x}) > l)f(\boldsymbol{x})}{P(z > l)}.$$

11.3.2 Stochastic Importance Sampling Method 2

Choe et al. [37] propose an alternative stochastic importance sampling-based estimator that restricts N_i to one, such that

$$\hat{P}_{\text{SIS2}}(z > l) = \frac{1}{N_T} \sum_{i=1}^{N_T} \mathbb{1}(g(\boldsymbol{x}_i) > l)\mathcal{L}(\boldsymbol{x}_i). \tag{11.16}$$

Although the right-hand side of Eq. 11.16 looks the same as that in Eq. 11.6, a profound difference is that $g(\cdot)$ function here is not deterministic. As a result, q_{DIS} cannot be used as q_{SIS2}. Choe et al. [37] present the optimal density function as

$$q^*_{\text{SIS2}}(\boldsymbol{x}) = \frac{1}{C_{q2}} f(\boldsymbol{x})\sqrt{S(\boldsymbol{x})}, \tag{11.17}$$

where C_{q2} is another normalizing constant such that

$$C_{q2} = \int f(\boldsymbol{x})\sqrt{S(\boldsymbol{x})}d\boldsymbol{x}.$$

The importance sampling density, q^*_{SIS2}, also reduces to q^*_{DIS} when the stochastic simulator is replaced by a deterministic simulator. Again, under a deterministic simulator, $S(\boldsymbol{x}) = \mathbb{1}(g(\boldsymbol{x}) > l)$, i.e., an indicator function taking either zero or one. Therefore, $\sqrt{S(\boldsymbol{x})} = S(\boldsymbol{x})$, so that the density function becomes

$$q^*_{\text{SIS2}} = \frac{f(\boldsymbol{x})S(\boldsymbol{x})}{\int f(\boldsymbol{x})S(\boldsymbol{x})d\boldsymbol{x}} = \frac{\mathbb{1}(g(\boldsymbol{x}) > l)f(\boldsymbol{x})}{P(z > l)}.$$

11.3.3 Benchmark Importance Sampling Method

Choe et al. [37] mimic the deterministic importance sampling density function by replacing the failure-verifying indicator function in Eq. 11.6 with the conditional POE, and call the resulting importance sampling density function the benchmark importance sampling (BIS) density, i.e.,

$$q^*_{\text{BIS}}(\boldsymbol{x}) = \frac{S(\boldsymbol{x})f(\boldsymbol{x})}{P(z > l)} = \frac{S(\boldsymbol{x})f(\boldsymbol{x})}{\int S(\boldsymbol{x})f(\boldsymbol{x})d\boldsymbol{x}},$$

and use Eq. 11.6 as the failure probability estimator. To be consistent with the notations used in q^*_{SIS1} and q^*_{SIS2}, we denote by C_{q_B} the normalizing constant in the above density function, i.e.,

$$q^*_{\text{BIS}}(\boldsymbol{x}) = \frac{1}{C_{q_B}} f(\boldsymbol{x})S(\boldsymbol{x}), \tag{11.18}$$

where,

$$C_{q_B} = \int f(\boldsymbol{x})S(\boldsymbol{x})d\boldsymbol{x}.$$

11.4 IMPLEMENTING STOCHASTIC IMPORTANCE SAMPLING

In the stochastic importance sampling densities, described in the preceding section, two pieces of detail need to be sought out for their implementation. The first is about modeling the conditional POE, $S(\boldsymbol{x})$, and the other is how to sample from a resulting importance sampling density without necessarily computing the normalized constant in the denominator.

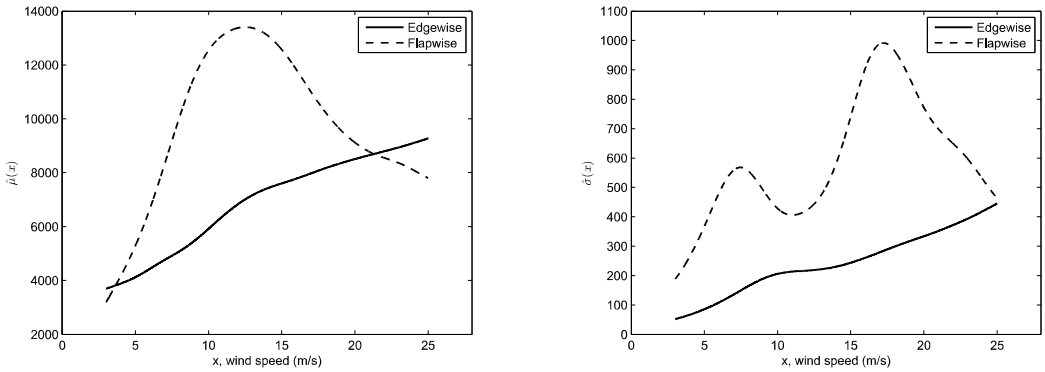
11.4.1 Modeling the Conditional POE

Choe et al. [37] suggest using a meta-modeling approach to establish an approximation for $S(\boldsymbol{x})$, but argue that using the Gaussian process model is appropriate when the modeling focus is on the part around the mode of a probability density function. For extreme load and failure probability analysis, the focus is instead on the extreme quantiles and the tail probability of a skewed distribution. Unlike Dubourg et al. [53] who use the Gaussian process-based approach, Choe et al. use a generalize additive model for location, scale, and shape (GAMLSS) [179].

In Chapter 10, a GEV distribution is used to model the extreme load on critical turbine components. The GEV distribution has three distribution parameters: location, scale, and shape. Section 10.4 presents an inhomogeneous GEV distribution, in which the location parameter and the scale parameter are modeled as a function of the input $\boldsymbol{x}$ using MARS models. The approach in Section 10.4 falls under the broad umbrella of GAMLSS.

In [37], Choe et al. still use a GEV distribution and also model its location and shape parameter as a function of the input, while keeping the shape parameter fixed, the same approach as used in Section 10.4. Choe et al. choose to include only the wind speed in $\boldsymbol{x}$, so that the functions for the location and shape parameters are univariate. For this reason, Choe et al. use a smoothing spline to model both functions, rather than the MARS function used in Section 10.4. Recall that a smoothing spline handles a univariate input well but does not scale very well in higher input dimensions. MARS is one popular multivariate spline-based models handling multi-dimensional inputs. Please visit Section 5.3.3 for more details on smoothing splines and spline-based regression.

Choe et al. [37] obtain a training dataset using the NREL simulators. The training dataset consists of 600 observation pairs of $\{x_i, y_i\}$, $i = 1, 2, \ldots, 600$. The x is the wind speed sampled from a Rayleigh distribution but truncated between the cut-in wind speed at 3 m/s and the cut-out wind speed at 25 m/s. The y is the corresponding load response obtained by running the NREL simulators. Slightly different from the smoothing spline formulation in Eq. 5.22, here are two smoothing splines, one for the location parameter and the other for the scale parameter, to be estimated simultaneously. Following the GAMLSS framework, Choe et al. maximize an objective function regularized by both smoothing splines. Let $\mu(x)$ be the location function, $\sigma(x)$ be



(a) Location function (b) Scale function

FIGURE 11.2 Location and scale parameter functions for both bending moments responses. (Reprinted with permission from Choe et al. [37].)

the scale function, and γ_μ and γ_σ be the two respective penalty parameters. The objective function is then

$$\min \left\{ \text{log-lik} - \gamma_\mu \int \mu''(t)^2 dt - \gamma_\sigma \int (\log \sigma(t)'')^2 dt \right\}, \tag{11.19}$$

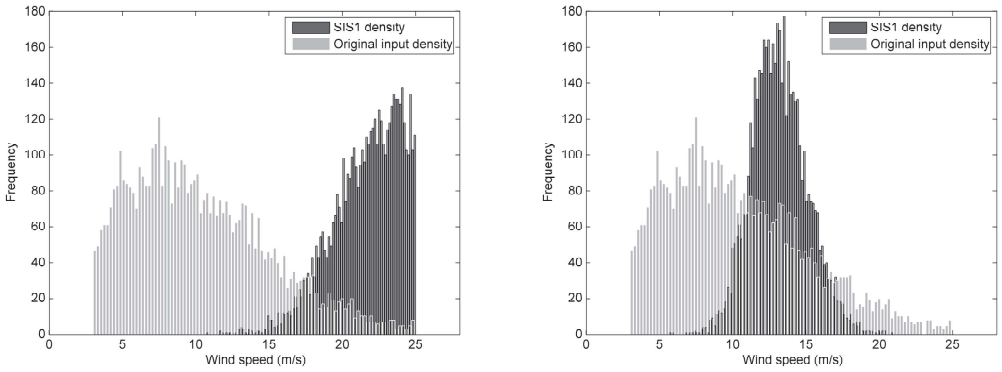
where log-lik refers to the log-likelihood function using the training dataset.

Fig. 11.2 presents the estimated functions for the location and scale parameters using the 600 data pairs in the training set. The shape parameter, kept constant in the above modeling process, is estimated at -0.0359 for the edgewise bending moments response and at -0.0529 for the flapwise bending moments response. In both cases, the resulting GEV distribution exhibits the pattern of a reverse Weibull distribution.

11.4.2 Sampling from Importance Sampling Densities

The three importance sampling densities, q^*_{SIS1}, q^*_{SIS2}, and q^*_{BIS} in Section 11.3, all have a normalizing constant in the denominator of their respective expression. Let us refer to this normalizing constant generically as C_q. Specifically, $C_q = C_{q1}$ in q^*_{SIS1}, $C_q = C_{q2}$ in q^*_{SIS2}, and $C_q = C_{q_B}$ in q^*_{BIS}. In order to compute the failure probability estimate using Eq. 11.12 or Eq. 11.16, these constants need to be numerically evaluated. All the constants involve the integration of one known function, $f(\boldsymbol{x})$, and a meta-model function, $S(\boldsymbol{x})$, so that a numerical integration routine can compute these constants. In their study [37], Choe et al. use the `MATLAB` function `quadgk` for the numerical integration whose input is the univariate wind speed. If one has multiple inputs in $\boldsymbol{x}$ and needs to use a numerical integrator for multivariate inputs, Choe et al. recommend using `mcint`.

For drawing samples from the importance density functions, Choe et



(a) Edgewise moments with $l = 9,300$ kNm (b) Flapwise moments with $l = 14,300$ kNm

FIGURE 11.3 Empirical SIS1 importance sampling density for both bending moments responses, overlaid on top of the density function of wind speed $f(x)$. (Reprinted with permission from Choe et al. [37].)

al. [37] skip the computing of these normalizing constant. They advocate using an acceptance-rejection algorithm to sample from the respective importance sampling density. The acceptance-rejection algorithm samples a u from a uniform distribution over the interval of $[0, f(\boldsymbol{x})]$ and then compares u with $C_q \cdot q^*(\boldsymbol{x})$. If u is smaller, then accept $\boldsymbol{x}$ as a valid sample; otherwise, reject this sample and repeat the sampling action and check again.

Note that the acceptance-rejection condition, $u \leq C_q \cdot q^*(\boldsymbol{x})$, does not involve computing C_q, because $C_q \cdot q^*(\boldsymbol{x})$ can be determined based on $f(\boldsymbol{x})$ and $S(\boldsymbol{x})$, according to Eqs. 11.15a, 11.17, and 11.18.

11.4.3 The Algorithm

Choe et al.'s algorithm to execute the importance sampling using stochastic simulators is summarized in Algorithm 11.2. Fig. 11.3 presents the empirical importance sampling densities of both bending moments responses, overlaid on top of the original wind speed density $f(x)$. The importance sampling densities in Fig. 11.3 are obtained by using q^*_{SIS1}. Similar results can be obtained by using either q^*_{SIS2} or q^*_{BIS}. One can observe from Fig. 11.3 that the distribution of samples over the wind spectrum is different under the importance sampling density versus that under the original wind distribution. Where the high mass of samples appears depends on the physical mechanism governing the bending moments response and exhibits close correlation with the trend shown in the respective plot in Fig. 11.1.

Algorithm 11.2 Importance sampling algorithm using stochastic simulators.

1. Approximate the conditional POE, $S(\boldsymbol{x})$, with a meta-model. In the case of turbine load response, estimate $S(\boldsymbol{x})$ using a small training dataset and fit an inhomogeneous GEV distribution model.

2. Select one of the stochastic importance sampling densities and obtain the set of samples, $\{\boldsymbol{x}_i,\ i = 1, \ldots, M\}$, based on the following acceptance-rejection procedure:

   (a) Sample $\boldsymbol{x}$ from the original input distribution, $f(\boldsymbol{x})$.

   (b) Sample a u from the uniform distribution over $[0, f(\boldsymbol{x})]$.

   (c) If $u \leq C_q \cdot q^*(\boldsymbol{x})$, return $\boldsymbol{x}$ as an sample drawn from the respective importance sampling density; otherwise, discard the sample and draw a new sample of $\boldsymbol{x}$ from $f(\cdot)$.

   (d) Repeat the acceptance-rejection check and sampling action until the prescribed sample size M is reached.

3. For SIS1, determine the allocation size, N_i^*, using Eq. 11.15b, for each $\boldsymbol{x}_i$. For SIS2 and BIS, $N_i^* = 1$.

4. Run the stochastic simulator N_i^* times at each $\boldsymbol{x}_i$, $i = 1, 2, \ldots, M$.

5. Estimate the failure probability using Eq. 11.12 or Eq. 11.16.

11.5 CASE STUDY

Choe et al. [37] present both a numerical analysis, illustrating various aspects of the stochastic importance sampling method, and a case study, using the NREL simulator's responses to estimate the failure probability and to demonstrate the computational benefit of using the importance sampling method.

11.5.1 Numerical Analysis

In the numerical analysis, Choe et al. [37] use the following data generating mechanism

$$\begin{aligned} x &\sim \mathcal{N}(0,1), \\ y|x &\sim \mathcal{N}(\mu(x), \sigma^2(x)), \end{aligned} \tag{11.20}$$

where $\mu(x)$ and $\sigma(x)$ in the distribution of y are functions of input x. Specifically, $\mu(x)$ and $\sigma(x)$ are chosen as

$$\begin{aligned} \mu(x) &= 0.95\delta x^2(1 + 0.5\cos(5x) + 0.5\cos(10x)), \quad \text{and} \\ \sigma^2(x) &= 1 + 0.7|x| + 0.4\cos(x) + 0.3\cos(14x). \end{aligned} \tag{11.21}$$

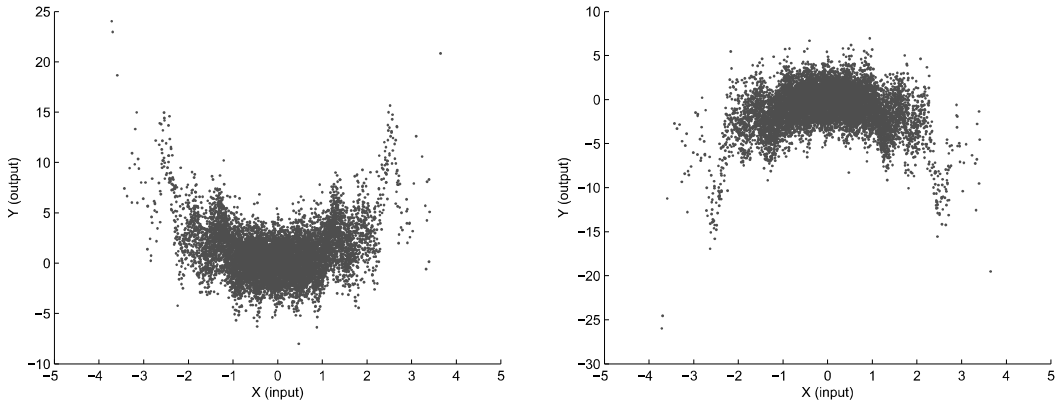
To use the stochastic importance sampling densities in Section 11.3, Choe et al. [37] specify the meta-models used for $\mu(x)$ and $\sigma(x)$, respectively, as

$$\begin{aligned} \hat{\mu}(x) &= 0.95\delta x^2(1 + 0.5\rho\cos(5x) + 0.5\rho\cos(10x)), \quad \text{and} \\ \hat{\sigma}(x) &= 1 + 0.7|x| + 0.4\rho\cos(x) + 0.3\rho\cos(14x), \end{aligned} \tag{11.22}$$

which are nearly the same as the location and scale functions in Eq. 11.21 but with a ρ inserted to control the accuracy of meta-modeling between $\mu(x)$ and $\hat{\mu}(x)$ and between $\sigma(x)$ and $\hat{\sigma}(x)$. Both $\mu(x)$ and $\hat{\mu}(x)$ also include a δ to control the similarity between the importance sampling density and the original density function of x.

The simulation parameters are set as $N_T = 1,000$ and $M = 300$ when using SIS1 or simply $N_T = 1,000$ for using SIS2 and BIS. To assess the uncertainty of the failure probability estimates, the numerical experiment is repeated 500 times so as to compute the standard error of a failure probability estimate. The computational efficiency is measured by the relative computational ratio of $N_T/N_T^{(\mathrm{CMC})}$, where $N_T^{(\mathrm{CMC})}$ is the total number of simulation runs required by a crude Monte Carlo method to achieve a standard error comparable to that achieved by using the importance sampling method.

The first numerical analysis sets $\rho = 1$, while choosing $\delta = 1$ or $\delta = -1$, and running for three failure probabilities, $P = 0.1$, 0.05, or 0.01. The analysis outcome is presented in Table 11.1. Choe et al. [37] observe that the computational benefit of using the stochastic importance sampling, as indicated by a small relative computational ratio, is more pronounced when the target probability is smaller, which is a desired property for the importance sampling method.



(a) $\delta = 1$ (b) $\delta = -1$

FIGURE 11.4 Sample scatter plots under different δ's. (Reprinted with permission from Choe et al. [37].)

The parameter δ affects the critical region where the importance sampling density function is supposed to draw its samples. In this simulation study, the critical region is where the large positive y values can be found. When $\delta = 1$, the critical region is where $|x|$ is large, i.e., at both ends of the input area and far away from the origin. This choice of δ thus makes the importance sampling density different from the original density of x, as the original density centers at zero. The choice of $\delta = -1$ flips the spread of samples vertically. Consequently, the critical region under $\delta = -1$ is around the origin, so that the resulting importance sampling density function has a great overlap with the original density function of x. In other words, $\delta = -1$ makes the importance sampling density less different from the original density. To appreciate this effect, please see the sample scatter plots in Fig. 11.4, drawn with $\delta = 1$ and $\delta = -1$, respectively. Note how much in each case, or how much less, the positive tails overlap with the area around the origin.

When the importance sampling density is different from the original density, the computational gain by using the importance sampling method is supposed to be more substantial. This is confirmed by the analysis outcome in Table 11.1, where the computational benefit is greater when $\delta = 1$ than that when $\delta = -1$.

In the second analysis, Choe et al. [37] vary ρ in $\hat{\mu}(x)$ and $\hat{\sigma}(x)$ so that the meta-model may deviate from the respective true function. Table 11.2 presents the analysis result for $\rho = 1$, 0.50, and 0. The standard error of the failure probability estimate does increase as ρ decreases, but the rate of increase for SIS1 and SIS2 is slower than that for BIS. The slowest increase is witnessed in the case of SIS2, whose standard error increases about 67% from a perfect meta-model (when $\rho = 1$) to a meta-model substantially different from the original model (when $\rho = 0$), whereas the standard error increases three times

TABLE 11.1 Estimates of the failure probability and the associated standard errors ($\rho = 1$).

		$\delta = 1$		
		$P = 0.10$	$P = 0.05$	$P = 0.01$
SIS1	Average estimate	0.1004	0.0502	0.0100
	Standard error	0.0068	0.0039	0.0005
	$N_T/N_T^{(\mathrm{CMC})}$	51%	32%	2.5%
SIS2	Average estimate	0.0999	0.0501	0.0100
	Standard error	0.0069	0.0042	0.0006
	$N_T/N_T^{(\mathrm{CMC})}$	53%	37%	3.6%
BIS	Average estimate	0.1002	0.0505	0.0101
	Standard error	0.0089	0.0068	0.0014
	$N_T/N_T^{(\mathrm{CMC})}$	88%	97%	20%
CMC	Average estimate	0.1005	0.0506	0.0100
	Standard error	0.0092	0.0070	0.0030
		$\delta = -1$		
		$P = 0.10$	$P = 0.05$	$P = 0.01$
SIS1	Average estimate	0.1001	0.0500	0.0100
	Standard error	0.0090	0.0062	0.0026
	$N_T/N_T^{(\mathrm{CMC})}$	90%	81%	68%
SIS2	Average estimate	0.1001	0.0500	0.0099
	Standard error	0.0086	0.0064	0.0028
	$N_T/N_T^{(\mathrm{CMC})}$	82%	86%	79%
BIS	Average estimate	0.1009	0.0503	0.0101
	Standard error	0.0095	0.0067	0.0031
	$N_T/N_T^{(\mathrm{CMC})}$	100%	95%	97%
CMC	Average estimate	0.1005	0.0498	0.0100
	Standard error	0.0096	0.0071	0.0031

Source: Choe et al. [37]. With permission.

TABLE 11.2 Effect of ρ on failure probability estimate ($\delta = 1$ and $P = 0.01$).

		ρ		
		1.00	0.50	0
SIS1	Average estimate	0.0100	0.0100	0.0101
	Standard error	0.0005	0.0008	0.0017
SIS2	Average estimate	0.0100	0.0101	0.0100
	Standard error	0.0006	0.0007	0.0010
BIS	Average estimate	0.0101	0.0100	0.0102
	Standard error	0.0014	0.0018	0.0063
CMC	Average estimate	0.0099	0.0099	0.0099
	Standard error	0.0030	0.0030	0.0030

Source: Choe et al. [37]. With permission.

in the case of SIS1 and four and a half times in the case of BIS. Choe et al. state that SIS2 is less sensitive to the quality of meta-modeling, making SIS2 a robust, and thus favored, choice in the applications of importance sampling. While the standard errors of SIS1 and SIS2 remain substantially smaller than that of CMC even when $\rho = 0$, the standard error of BIS grows exceeding, and in fact, more than doubling, that of CMC at $\rho = 0$, indicating that the approach disregarding the stochasticity in a stochastic simulator's response has serious drawbacks.

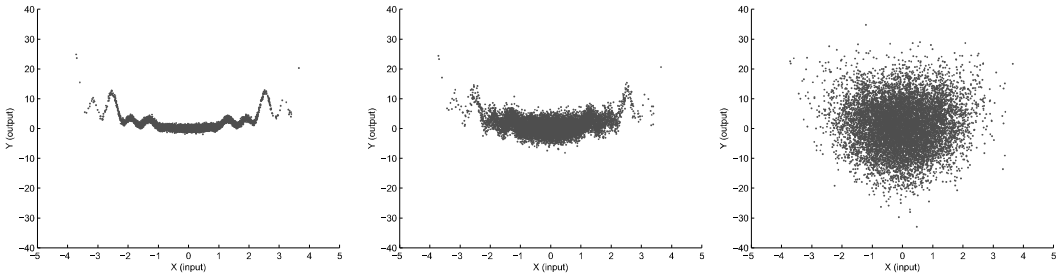
Recall that the deterministic importance sampling density in Eq. 11.8 leads to a failure probability estimate of zero variance. It is also mentioned in Sections 11.3.1 and 11.3.2 that when the response of a stochastic simulator becomes less variable under a given input, the two stochastic importance sampling densities reduce to a deterministic importance sampling density. Putting the two pieces of information together, one expects to see failure probability estimates of much smaller standard errors when SIS1 and SIS2 are used on less variable stochastic simulators.

To show this effect, Choe et al. [37] devise a numerical experiment in their third analysis, in which they change $\sigma(x)$ in Eq. 11.21 to

$$\sigma^2(x) = \tau^2, \tag{11.23}$$

while keeping $\mu(x)$ unchanged. Choe et al. vary τ to control the variability in the response of the simulator. Fig. 11.5 visualizes the variability in response under three values of τ. Comparing the spread of data samples for a given x value demonstrates that the variability in the response when $\tau = 0.5$ is much smaller than that when $\tau = 8$.

Table 11.3 presents the failure probability estimates and the associated



(a) $\tau = 0.5$ (b) $\tau = 2$ (c) $\tau = 8$

FIGURE 11.5 Sample scatter plots under different τ's. (Reprinted with permission from Choe et al. [37].)

TABLE 11.3 Effect of randomness in the simulator's response on the failure probability estimate ($\rho = 1$, $\delta = 1$ and $P = 0.01$).

		τ				
		0.50	1.00	2.00	4.00	8.00
SIS1	Average estimate	0.0102	0.0101	0.0101	0.0102	0.0100
	Standard error	0.0001	0.0001	0.0005	0.0021	0.0028
SIS2	Average estimate	0.0102	0.0101	0.0101	0.0104	0.0100
	Standard error	0.0001	0.0002	0.0006	0.0023	0.0028

Source: Choe et al. [37]. With permission.

standard errors. It is evident that when τ gets smaller, the standard errors of the failure probability estimates, resulting from both stochastic importance sampling methods, get close to zero quickly.

11.5.2 NREL Simulator Analysis

Choe et al. [37] employ the stochastic importance sampling method to estimate the failure probability using the NREL simulators. Both edgewise bending moments and flapwise bending moments are studied. There are two design load levels used, which are $l = 8,600$ kNm and $l = 9,300$ kNm for edgewise bending moments, and $l = 13,800$ kNm and $l = 14,300$ kNm for flapwise bending moments. The two load levels are chosen so that they correspond roughly to the failure probability of $P = 0.05$ and $P = 0.01$, respectively. The total computational runs set for the two design levels are, respectively, $N_T = 1,000$ and $N_T = 3,000$ for the edgewise bending moments response and $N_T = 2,000$ and $N_T = 9,000$ for the flapwise bending moments response. When using the same number of computational runs, the average estimates of the failure probability by the three importance sampling methods are comparable but their standard errors are different. Using SIS1 leads to the smallest standard

error, whereas using BIS sees a sizeable increase in the resulting standard error.

To assess the computation required for the crude Monte Carlo method to attain the same level of estimation accuracy, one could run the simulators a sufficient number of times, as one has run the simulator under the importance sampling method. Running the NREL simulator takes about one minute, not much for a single run. The difficulty is that a crude Monte Carlo method needs sometimes more than 60,000 runs of simulation to attain the same level of estimation accuracy as the importance sampling method does. Sixty thousand NREL simulator runs would take more 40 days to complete, too time consuming to be practical. For this reason, $N_T^{(\mathrm{CMC})}$ is computed by using Eq. 11.3 without actually running the NREL simulators under CMC. To compute N_T, one plugs in the standard error attained by the importance sampling method and the target probability value, P (note that in Eq. 11.3, $Var[\hat{P}_{\mathrm{CMC}}]$ is the square of the standard error). Taking the edgewise bending moments as an example, CMC needs about 11,000 runs to attain the same estimation accuracy attained in 1,000 runs for $l = 8,600$ kNm by the importance sampling method using SIS2, or about 51,000 runs for $l = 9,300$ kNm, as compared to 3,000 runs needed by the importance sampling method using SIS2. When compared with the importance sampling method using SIS1, the two run numbers become $18,000$ and $61,000$, respectively.

Tables 11.4 and 11.5 present, respectively, the failure probability estimates for edgewise and flapwise bending moments. In the tables, the standard error is computed by repeating the computer experiments 50 times. The 95% confidence intervals of the standard error are obtained by running a bootstrap resampling and using the bootstrap percentile interval [55]. In general, SIS1 performs the best but SIS2 performs rather comparably. Both SIS1 and SIS2 outperform BIS by a noticeable margin. Both SIS1 and SIS2 use only a fraction of simulation runs that would be needed by CMC in the case of estimating the failure probability for edgewise bending moments. The computational benefit in the case of flapwise bending moments is not as pronounced as in the case of edgewise bending moments, primarily because the importance sampling densities are not as much different from the original density $f(\cdot)$ in the case of flapwise bending moments. Still, even for flapwise bending moments, the computation needed by SIS1 is only about one-third of what is needed for CMC.

It is interesting to observe the appreciable difference between the stochastic importance sampling methods and BIS, especially between SIS2 and BIS. Looking at Eqs. 11.17 and 11.18, one notices that the density functions are rather similar. The normalizing constants are different, but that difference does not affect the sampling process outlined in Algorithm 11.2. The essential difference is in the numerator, where SIS2 uses a $\sqrt{S(\boldsymbol{x})}$, while BIS uses $S(\boldsymbol{x})$ without taking the square root. That simple action apparently makes a profound difference, as SIS2 is more efficient and requires fewer simulation runs than BIS does, for achieving a comparable standard error. Comparisons

TABLE 11.4 Estimates of the failure probability and the associated standard errors for edgewise bending moments.

Method	$l = 8,600, N_T = 1,000$		
	Average estimate	Standard error (95% bootstrap CI)	$N_T/N_T^{(\mathrm{CMC})}$
SIS1	0.0486	0.0016 (0.0012, 0.0020)	5.5%
SIS2	0.0485	0.0020 (0.0016, 0.0024)	8.7%
BIS	0.0488	0.0029 (0.0020, 0.0037)	18%
Method	$l = 9,300, N_T = 3,000$		
	Average estimate	Standard error (95% bootstrap CI)	$N_T/N_T^{(\mathrm{CMC})}$
SIS1	0.00992	0.00040 (0.00032, 0.00047)	4.9%
SIS2	0.01005	0.00044 (0.00036, 0.00051)	5.9%
BIS	0.00995	0.00056 (0.00042, 0.00068)	9.6%

Source: Choe et al. [37]. With permission.

TABLE 11.5 Estimates of the failure probability and the associated standard errors for flapwise bending moments.

Method	$l = 13,800, N_T = 2,000$		
	Average estimate	Standard error (95% bootstrap CI)	$N_T/N_T^{(\mathrm{CMC})}$
SIS1	0.0514	0.0028 (0.0022, 0.0033)	32%
SIS2	0.0527	0.0032 (0.0025, 0.0038)	42%
BIS	0.0528	0.0038 (0.0030, 0.0044)	59%
Method	$l = 14,300, N_T = 9,000$		
	Average estimate	Standard error (95% bootstrap CI)	$N_T/N_T^{(\mathrm{CMC})}$
SIS1	0.01070	0.00061 (0.00047, 0.00074)	32%
SIS2	0.01037	0.00063 (0.00046, 0.00078)	34%
BIS	0.01054	0.00083 (0.00055, 0.00110)	59%

Source: Choe et al. [37]. With permission.

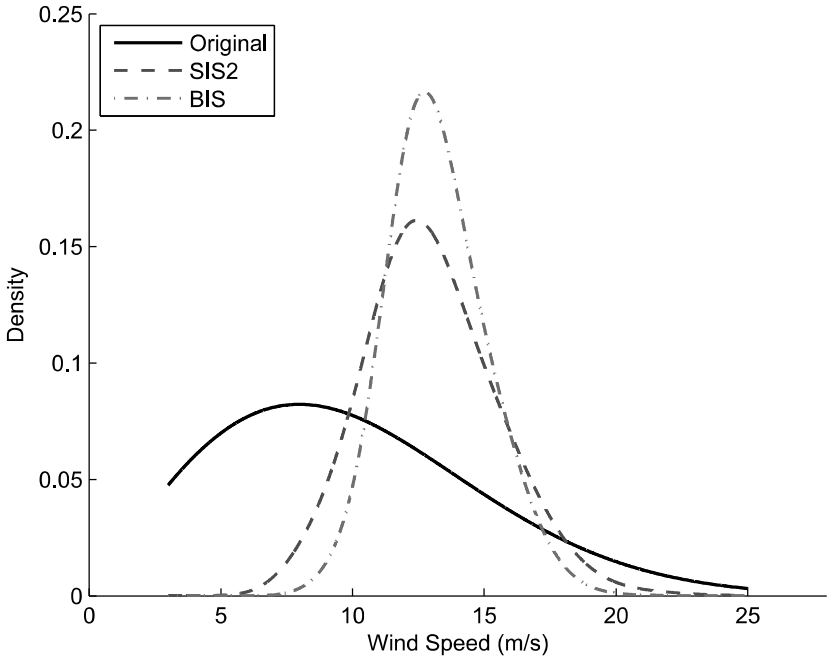


FIGURE 11.6 Comparison of three density functions: the original density function of wind speed, $f(\cdot)$, the SIS2 density function, and the BIS density function.

presented in Table 11.2 also show that SIS2 is more robust than BIS against, or less sensitive to, the meta-model's misspecification.

Fig. 11.6 presents a comparison of the two density functions. The original density function of wind speed, $f(\cdot)$, is also shown in Fig. 11.6. Not surprisingly, the concentration of $f(\cdot)$ does not coincide with the critical region. It turns out that BIS is able to focus on the correct sampling region. As compared with SIS2, however, BIS's focus is a bit too narrow and that action back fires. The square-root operation used in SIS2 appears crucial to attain the right balance in the bias (where to focus) versus variance (how narrowly to focus) tradeoff.

One more note is regarding the level of the target failure probability used in the case study. In Section 11.2.1, we cite a target failure probability at the level of 10^{-5}, but in the case study, the target probability is at or larger than 0.01. The reason that a smaller probability is not used is because doing so demands many more simulation runs than a numerical analysis could tolerate. Consider an importance sampling method that uses one percent of the runs required by CMC, for achieving the desired estimation accuracy for a target failure probability of $P = 10^{-5}$. As calculated in Section 11.2.1, CMC would need 6.7×10^7 simulation runs, and one percent of the CMC simulations is to run the simulator 6.7×10^5 times. When each simulator run takes one minute,

as opposed to one second, it takes more than a year to run the simulator that many times. In practice, in order to estimate the small failure probability for a turbine's 20-year or 50-year service, an iterative procedure, in the spirit of Algorithm 11.1, together with parallel computation taking advantage of multiple CPU cores, is inevitable. These solution approaches are in fact being actively pursued in ongoing research.

GLOSSARY

BIS: Benchmark importance sampling

CI: Confidence interval

CMC: Crude Monte Carlo

CPU: Central processing unit

DEVS: Discrete event system specification

DIS: Deterministic importance sampling

EOI: Events of interest

GAMLSS: Generalized additive model for location, scale, and shape

GEV: Generalized extreme value

IEC: International Electrotechnical Commission

IS: Importance sampling

MARS: Multivariate adaptive regression splines

NREL: National renewable energy laboratory

POE: Probability of exceedance

SIS: Stochastic importance sampling

EXERCISES

11.1 Prove the expectation and variance formulas in Eq. 11.3, which are about a crude Monte Carlo method's ability to estimate a failure probability.

11.2 Derive Eq. 11.7, the variance expression for the importance sampling method using a deterministic computer simulator.

11.3 The Kullback-Leibler distance between a pair of density functions, $g(\cdot)$ and $h(\cdot)$, is defined as

$$\mathcal{D}(g,h) = \mathbb{E}_g\left[\ln\frac{g(\boldsymbol{x})}{h(\boldsymbol{x})}\right] = \int g(\boldsymbol{x})\ln g(\boldsymbol{x})d\boldsymbol{x} - \int g(\boldsymbol{x})\ln h(\boldsymbol{x})d\boldsymbol{x}. \tag{P11.1}$$

The cross-entropy between the same two density functions is defined as

$$\mathcal{H}(g,h) = \mathbb{E}_g\left[-\ln h(\boldsymbol{x})\right] = -\int g(\boldsymbol{x}) \ln h(\boldsymbol{x}) d\boldsymbol{x}. \tag{P11.2}$$

The entropy function of $g(\cdot)$ is the cross-entropy between $g(\cdot)$ and itself, i.e., $\mathcal{H}(g,g)$.

a. Express the Kullback-Leibler distance using an entropy and a cross-entropy.

b. In Algorithm 11.1, our objective is to minimize the distance between $q^*(\boldsymbol{x})$ and $f(\boldsymbol{x},\boldsymbol{v})$, in order to choose $\boldsymbol{v}$, where q^* is the optimal importance sampling density to be solved for. Show that the minimization of $\mathcal{D}(q^*,f)$ is the same as maximizing $-\mathcal{H}(q^*,f)$, the negative cross-entropy between the two density functions. This is why the algorithm is referred to as a cross-entropy method.

c. Prove that Eq. 11.10 is meant to minimize the $\mathcal{D}(q^*,f)$ or maximize $-\mathcal{H}(q^*,f)$ (through their empirical counterparts).

11.4 Derive Eq. 11.14, the variance expression of the failure probability estimate, $\hat{P}_{\text{SIS1}}$.

11.5 Derive the optimal density, q^*_{SIS1}, and the optimal allocation, N_i^*, in Eq. 11.15.

11.6 Prove that the variance of $\hat{P}_{\text{SIS2}}$ takes the following expression.

$$Var\left[\hat{P}_{\text{SIS2}}\right] = \frac{1}{N_T}\left(\mathbb{E}_f\left[S(\boldsymbol{x})\mathcal{L}(\boldsymbol{x})\right] - P(z>l)^2\right). \tag{P11.3}$$

11.7 Derive the optimal density, q^*_{SIS2}, in Eq. 11.17.

11.8 Using the data pairs in the training set, build a kriging-based meta-model, $S(\boldsymbol{x})$. For this purpose, please use the ordinary kriging model in Eq. 3.8 without the nugget effect. Establish a kriging meta-model for the edgewise bending moments response and another for the flapwise bending moments response.

11.9 Using the meta-models created in Exercise 11.8 and draw wind speed samples using the importance sampling density functions. Plot the empirical distribution of the resulting samples and overlay them on top of the original wind speed samples; the same is done in Fig. 11.3. Observe the empirical distributions and compare them with their counterparts in Fig. 11.3.

11.10 Let us modify the location function and the associated meta-model in Eqs. 11.21 and 11.22 to the following:

$$\begin{aligned} \mu(x) &= 0.95x^2(1 + 0.5\cos(10\nu x) + 0.5\cos(20\nu x)), \quad \text{and} \\ \hat{\mu}(x) &= 0.95\beta x^2(1 + 0.5\cos(10\nu x) + 0.5\cos(20\nu x)), \end{aligned} \tag{P11.4}$$

where β is the scaling difference between the two functions, while ν controls the roughness of the location function. When $\nu = 0$, the location function and its meta-model reduce to a quadratic function of x.

a. Set the target failure probability $P = 0.01$ and the roughness parameter $\nu = 0.5$. Investigate the effect of β on the failure probability estimate. Try for $\beta = 0.90, 0.95, 1.00, 1.05,$ and 1.10. For each of the β values, produce the average and standard error of the failure probability estimate for four methods, SIS1, SIS2, BIS, and CMC.

b. Set $P = 0.01$ and $\beta = 1$. Investigate the effect of ν on the failure probability estimate. Try for $\nu = 0, 0.50,$ and 1.00. Same as in part (a), produce the average and standard error of the failure probability estimate for four methods, SIS1, SIS2, BIS, and CMC.

CHAPTER 12

Anomaly Detection and Fault Diagnosis

Load assessment, as introduced in Chapters 10 and 11, definitely plays an important role in wind turbine reliability management. But load assessment addresses a specialized category of problems and faults, which happen as a result of excessive mechanical load. A wind turbine generator is a complex system, comprising a large number of electro-mechanical elements. Many other types of operational anomalies and faults could happen and do take place. This is the reason that we dedicate the last chapter to the general topic of anomaly detection and fault diagnosis. Anomaly detection techniques are supposed to identify anomalies from loads of seemingly homogeneous data and lead analysts and decision makers to timely, pivotal and actionable information. It bears a high relevance with the mission of reliability management for wind turbines.

In this chapter, we could not run the case study using wind turbine fault data. Instead, the methods introduced in the chapter are demonstrated using a group of publicly accessible benchmark datasets, plus a hydropower plant dataset.

12.1 BASICS OF ANOMALY DETECTION

12.1.1 Types of Anomalies

Loosely speaking, anomalies, also referred to as outliers, are data points or a cluster of data points which lie away from the neighboring points or clusters and are inconsistent with the overall pattern of the data. A universal definition of anomaly is difficult to come by, as what constitutes an anomaly often depends on the context.

Goldstein and Uchida [77] illustrate a few different types of anomalies; please see Fig. 12.1. Points A_1 and A_2 are referred to as the global point-

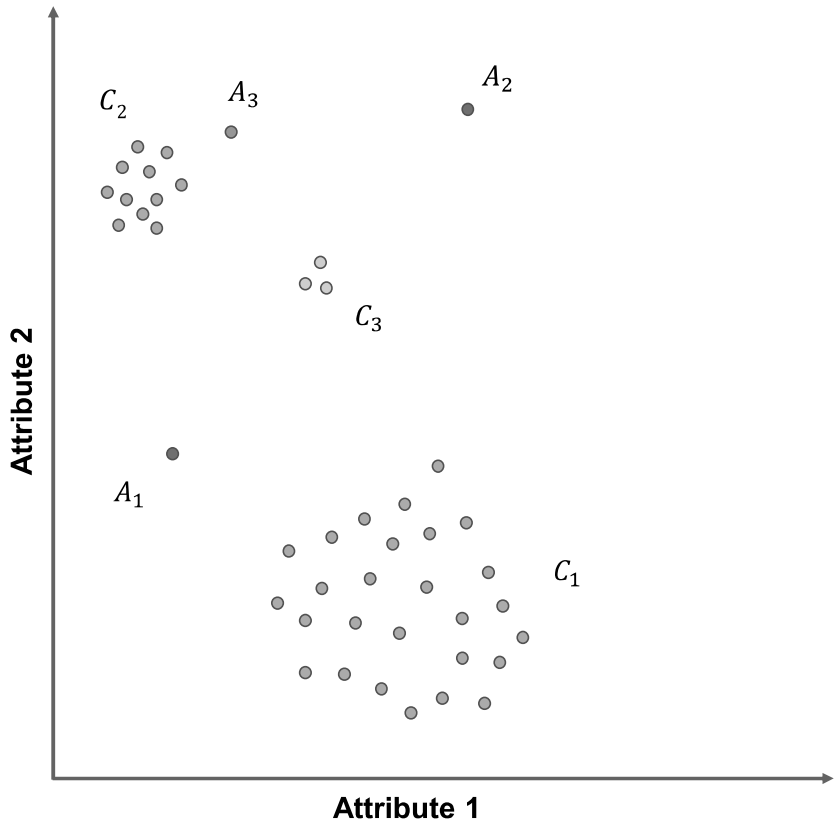


FIGURE 12.1 Illustration of different types of anomalies. (Source: Goldstein and Uchida [77].)

wise anomalies, as they are far away from the existing data points and data clusters. Point A_3 is referred to as a local pointwise anomaly, because, globally, this point is closer to the data cluster C_2 than to many other data points and data clusters, but locally and relative to C_2, it is away from the rest of data points in that cluster. The data clusters, C_2 (including A_3) and C_3, are considered as the collective anomalies or anomalous clusters, referring to the situation when a whole set of data behaves differently from the other regular clusters or data points. Of course, one can argue that should we deem C_2 to represent the normal operation conditions, the cluster, C_1, is the anomalous cluster. That is certainly possible. Absent strong prior knowledge suggesting otherwise, however, the anomalies are always treated as the minority cases in a dataset. Such treatment makes intuitive sense for especially engineering systems, because if the faults and anomalies become more numerous than the supposedly normal operation conditions, the said system is definitely in need of a redesign or an overhaul.

The complexity in defining anomalies translates to the challenges faced in anomaly detection. Methods for detecting anomalous clusters have a strong connection with the research in the field of clustering [86, Section 14.3] and

community detection [66, 154]. But in this chapter, our main focus is on the pointwise anomalies.

A branch of research relevant to anomaly detection is the field of statistical process control (SPC) or statistical quality control (SQC) [147]. SPC considers a time series response from a process that has a natural, underlying randomness and aims to detect a change that is deemed substantially greater than the inherent fluctuation of the underlying process. SPC methods are usually manifested in a control chart, a run chart with the horizontal axis representing the time and the vertical axis representing the response or a test statistic computed out of the response.

Fig. 12.2 presents two types of anomalies often encountered in the SPC literature. The left-hand plot of Fig. 12.2 shows a spike type of change. The right-hand plot shows a sustained mean shift—in this particular case, the process response increases substantially at time t_0 and stays there for the next period of time, until a new change happens in the future. This time instance, t_0, is called a *change point*, and for this reason, SPC methods are part of the change-point detection methods.

The two types of changes correspond naturally to the pointwise anomalies and the clustered anomalies. Apparently, the spike type of change is a pointwise anomaly, whereas the sustained mean shift partitions the dataset into two clusters of different operational conditions, one being normal and the other being anomalous. To represent both types of change detection, the term anomaly detection, also labeled as novelty detection or outlier detection, is often used together with the term change detection, creating the expression *change and anomaly detection*. The subtle difference between change detection and anomaly detection lies in the different types of changes or anomalies to be detected.

To detect a change or an anomaly, the SPC approaches rely on a statistical hypothesis test to decide if a mean shift or a spike comes from a statistically different distribution. The distribution in question is typically assumed to be, or approximated by, a Gaussian distribution. The detection mission can thus be reduced to detecting whether the key distribution parameter, either the mean or the variance (or both), is different, with a degree of statistical significance, from that of the baseline process. A control chart runs this statistical hypothesis test repeatedly or iteratively over time, comparing the newly arrived observations with the control limits, i.e., the decision limits, that are established according to the chosen degree of statistical significance, and triggering an anomaly alarm when something exceeds the control limits.

12.1.2 Categories of Anomaly Detection Approaches

Goldstein and Uchida [77] categorize anomaly detection approaches in three broad branches, depending on the availability and labeling of the data in a training set.

The first category is *supervised anomaly detection*, when one has appropri-

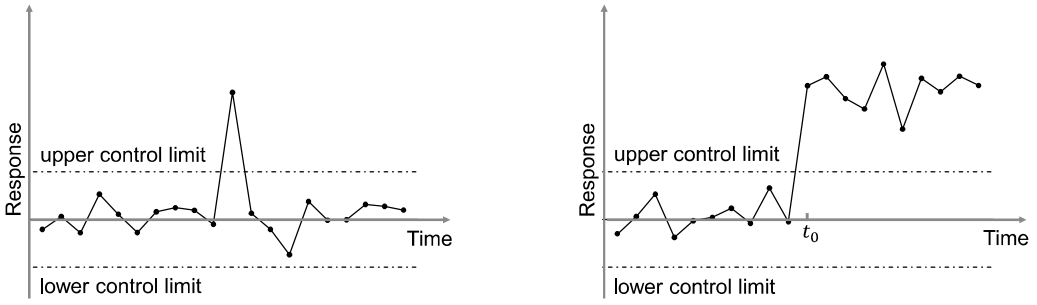


FIGURE 12.2 Control charts, change-point detection, and two types of anomalies.

ately labeled training data in advance, both normal and anomalous instances, so that analysts can train a model based on these labeled data and use the model to decide the labels of future data. This is in fact the classical two-class classification approach, and many of the data science methods introduced in the early part of this book can be used for this purpose, including SVM, ANN, kernel regression and classification, MARS, CART, among others. While using the two-class approaches for anomaly detection, analysts need to be mindful of the imbalance nature of the data instances in the training set. Understandably, the anomalies are far fewer than the normal instances. This data imbalance issue has received considerable attention and is still an active research topic [29, 35, 167].

The second category is known as *semi-supervised anomaly detection*, in which one has only the normal instances and no anomalous data. The idea is to employ the normal data to train a model and create a decision boundary enclosing the normal data. The approach classifies future observations as anomalies if they fall outside the decision boundaries. In other words, the semi-supervised anomaly detection is to define what constitutes the normalcy and treats anything that deviates from the normalcy as anomalies. One-class SVM [196] falls under this category. Park et al. [159] develop a non-parametric semi-supervised anomaly detection method which is proven to be asymptotically optimal. Park et al. show that under the Gaussianity assumption, their optimal detection method reduces to a Hotelling's T^2, a popular method used in SPC [147, Section 11.3].

The most difficult scenario is the absence of any labels for the data or the inability to assume that all data points are normal. As a result, it is not possible to conduct a supervised training. One therefore has to rely entirely on the structure of the dataset and to detect the anomalies in an unsupervised manner. This last category is known as *unsupervised anomaly detection*.

The SPC methods are commonly considered as a method of semi-supervised anomaly detection, as the control limits used in the control charts are based on the normal condition data, known as the *in-control* data in

the SPC vernacular. But an SPC procedure usually starts with a dataset for which one cannot guarantee all data instances are normal. This creates the desire of separating change and anomaly detection into two stages: Phase I and Phase II. Phase I is to separate the anomalous cases from the normal majority, whereas Phase II uses the normal majority to establish a decision boundary, or control limits, to flag an incoming observation if it is deemed anomalous. In this sense, Phase I in an SPC procedure is unsupervised, while Phase II is semi-supervised.

Our focus in this chapter is unsupervised anomaly detection. The relevance of an unsupervised anomaly detection is evident not only to the wind industry but to many engineering systems, instrumented with various types of sensors on many components or subsystems. When a service and maintenance engineer suspects that there is a malfunction in a turbine, she/he extracts a dataset from the control system that contains the collected sensor data for that turbine for a selected period of time (weeks, months, or even years), and then stores the data in a relational database or simply in a CSV file for further analysis. Staring at the spreadsheet of data, a service and maintenance engineer often wonders if there is an automated, efficient way to isolate the anomalies from the rest of the data. The historical data in the spreadsheet have almost surely both normal condition data and anomalies. It is just that the service and maintenance engineers do not know which is which. An unsupervised anomaly detection is meant to answer the call.

12.1.3 Performance Metrics and Decision Process

To assess the performance of an anomaly detection method, the usual type-I error versus type-II error trade-off applies. The type-I error, also known as false alarms or false positives, is when the underlying truth of the instance is normal, but the method nonetheless flags it as an anomaly. The type-II error, on the other hand, is when the underlying truth of the instance is an anomaly, but the method fails to flag it. The type-II error is also referred to as missed detections or false negatives. The trade-off between the two types of error says that with all other conditions and parameters held unchanged, one type of error can only be reduced at the expense of increasing the other type of error. Of course, it is possible to reduce both types of errors, but doing so calls for more capable methods or more resources like a larger sample size.

In the mission of anomaly detection, the desire to have a higher detection capability, or equivalently, a smaller type-II error, often triumphs a small type-I error. The fundamental reason is because an anomaly detection method is useful only if it can detect something. A method that rarely detects is utterly useless no matter how nice a property it has in terms of the false positive rate. In the meanwhile, if a detection method triggers too many false alarms, it will eventually become a nuisance and will be turned off in practice.

One common practice in maintaining a healthy trade-off between these two errors for anomaly detection is to set a cut-off threshold, say, N_o, and let an

anomaly detection method rank the data from being most likely anomalous to being least likely so. The top N_o ranked data instances are flagged as anomalies, whereas the rest are treated as normal. Once N_o is given, a commonly used performance metric is the precision at N_o ($P@N_o$) [30], defined as the proportion of correct anomalies identified in the top N_o ranks, such as

$$P@N_o = \frac{\#\{o_i \in \mathcal{O} \,|\, \text{rank}(o_i) \leq N_o\}}{N_o}, \tag{12.1}$$

where $\mathcal{O}$ is the set of true anomalies and o_i is the i-th element in the ranked dataset, according to their likelihood of being an anomaly. A small rank value implies a higher likelihood, so the most likely instance has a $\text{rank}(o_i) = 1$.

Under a given N_o, the goal is to have as high a $P@N_o$ as possible. When N_o is the number of true anomalies, the number of false positives or false alarms is simply $N_o - N_o \times P@N_o$. In reality, the number of true anomalies is not known. Still, a high detection rate at N_o strongly implies a lower false positive rate. For this reason, one does not always present the false positive rate separately.

Without knowing the number of true anomalies, one practical problem is how to set the cut-off threshold N_o. A good practice is to set N_o to be larger than the perceived number of anomalies but small enough to make the subsequent identification operations feasible. The rationale behind this choice lies in the fact that the false positive rate for anomaly detection problems is generally high, especially compared to the standard used for supervised learning methods. Despite a relatively high proportion of false positives, anomaly detection methods can still be useful, particularly used as a pre-screening tool. By narrowing down the candidate anomalies, it helps human experts a great deal to follow up with each circumstance and decide how to take a proper action or deploy a countermeasure. A fully automated anomaly detection is not yet realistic, due to the challenging nature of the problem. Therefore, a useful pre-screening tool, as the current anomaly detection offers, is valuable in filling the void, while analysts strive for the ultimate, full automation goal.

Not only is the number of true anomalies not known in practice, which data instance is a genuine anomaly is also unknown, as the dataset itself is unlabeled and finding out the anomalous instances is precisely what the method intends to do. Verifying the detection accuracy has to rely on another layer of heightened scrutiny, be it a more expensive and thus more capable detection instrument or method or a laborious and time-consuming human examination. In Section 12.6, we use a group of 20 benchmark datasets for which the true anomalies are known, plus a hydropower data for which the anomalies are verified manually by domain experts.

12.2 BASICS OF FAULT DIAGNOSIS

Detecting an anomaly is an important first step to inform proper actions to respond. Sometimes, the response or countermeasure needed could be obvious,

once the nature of the anomaly is revealed, but oftentimes, the anomaly just reveals the symptom of the problem. Yet multiple root causes may lead to the same symptom, so that a diagnostic follow-up is inevitable. This is very much analogous to medical diagnosis. A high body temperature and sore throat are anomalous symptoms on a healthy person. But a large number of diseases can cause these symptoms. Deciding what specific pathogen causes the symptoms is necessary before a proper medicine can be administrated to cure the illness.

Diagnosis of engineering systems relies heavily on the knowledge of the systems and know-how of their operations. The diagnostic process can hardly be fully automatic. Rather it is almost always human experts driven and could be labor intensive. But data science methods can facilitate the diagnostic process. For instance, a data science method can help find out which variables contribute to the anomalies and provide a pointed interpretation of each anomaly, thus aiding the domain expert to verify the root causes and fix them, if genuine. In this section, we present two commonly used diagnosis-aiding approaches: diagnosis based on supervised learning and visualization, and diagnosis based on signature matching.

12.2.1 Tree-Based Diagnosis

One immediate benefit of anomaly detection is that the outcomes of the detection can be used to convert the original unsupervised learning problem into a supervised learning problem. Suppose that the anomaly detection method does an adequate job, analysts can then label the data instances in the training set, according to their respective detection outcome. With the labeled dataset, many supervised learning methods can be used to extract rules or find out process variable combinations leading to the anomalous conditions.

The application of supervised learning methods is rather straightforward. While various methods can be used for this purpose, tree-based methods, like CART, are popular, due to its ability to visualize what leads to the anomalous outcomes. CART produces the learning outcomes in the fashion of mimicking a human-style decision-making process, which is another reason behind engineers' fondness of using this tool.

In the hydropower plant case, to be discussed in Section 12.6.2, after the anomalies are detected, a CART is built to facilitate the diagnosis process. While the bulk detail of that case study is to be discussed later, let us present the CART's learning outcome in Fig. 12.3.

From the resulting tree in Fig. 12.3, one can see that using the variables *Oil Temperature of Bearing 4*, *Air Pressure*, *Turbine Vertical Vibration* and *Delta Oil temp - Air Temp of Bearing 1* can correctly classify 25 anomalies based on the proper combination of their conditions. One such condition is when the oil temperature of bearing 4 is less than 27.216 degrees Celsius, the turbine generator almost surely behaves strangely. This condition consistently leads to eleven anomalous observations. Such specific information can certainly help domain experts go to the right components and subsystems

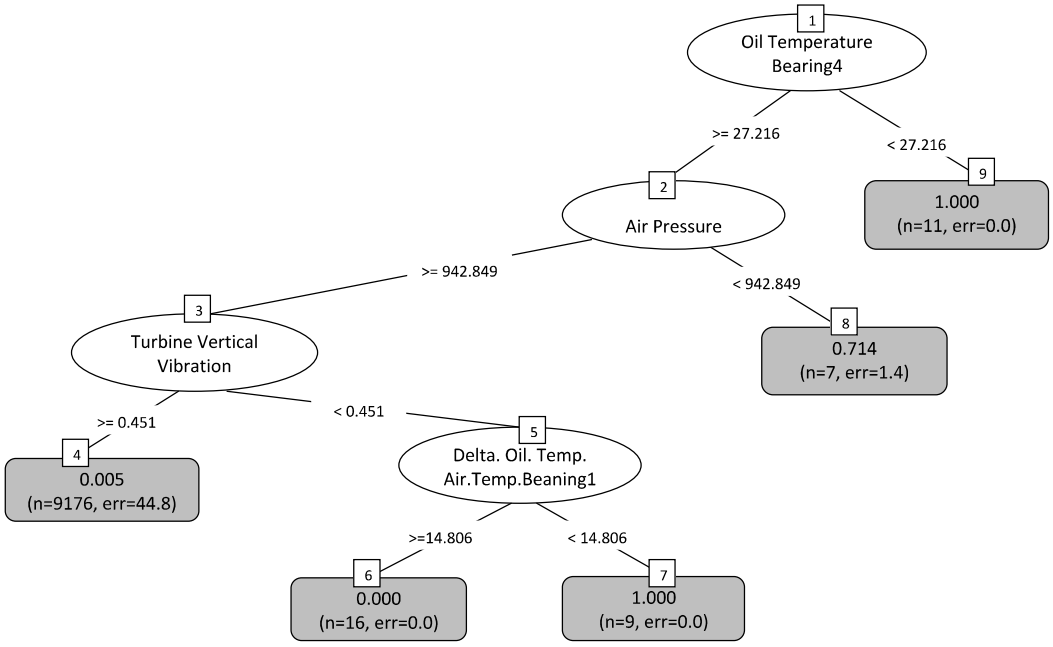


FIGURE 12.3 CART model based on anomaly detection and to be used to facilitate the diagnostic process. (Reprinted with permission from Ahmed et al. [4].)

to perform a follow-up and authenticate the root cause. The pertinent process conditions revealed by the CART model expedite the diagnostic process because the domain experts or the maintenance engineers save the effort of sifting through the large number of variables and data records to find and locate such conditions.

12.2.2 Signature-Based Diagnosis

The idea of signature-based diagnosis is intuitive. A signature library is built to store unique signatures of certain forms that have been associated with specific root causes or faults. If the data collected from ongoing operations can reveal the fault signature lurking in the current process, comparing the estimated signature with the ones in the signature library leads naturally to the identification of the responsible root cause, fulfilling the task of fault diagnosis.

While an intuitive idea, the specifics behind how to build the signature library and how to estimate the signature for ongoing operations can become involved. There is no universal definition of what constitutes a signature. A particular harmonics in the vibration signal resulting from gearbox rotations or the trace of a type of metallic ingredient in the lubricant oil can be signatures sought after. On the other hand, a signature may not be in plain sight

but needs to be worked out by building a mathematical model first. Invariably, the signature library building process is performed offline, while the signature estimation is conducted online, in a fashion similar to Fig. 9.4, in which one can replace "offline optimization" with "offline signature library" and "online control" by "online signature estimation."

The signature-based diagnosis approach has been successfully applied in many other industries, such as in the automotive assembly process [49]. The approach is general enough. With a similar model built for a wind turbine system, the method is applicable to the wind energy applications.

Let us briefly explain how the model-based signature matching works. For the sake of simplicity, let us consider a system whose input and output can be linked through a linear model, such as

$$\boldsymbol{y} = \mathbf{H}\boldsymbol{x} + \boldsymbol{\varepsilon}, \tag{12.2}$$

where $\boldsymbol{y}$ and $\boldsymbol{x}$ are the outputs and inputs, $\boldsymbol{\varepsilon}$ is assumed to be a Gaussian noise, and $\mathbf{H}$ is the system matrix capturing the input-output relationship. This equation in fact looks similar to the observation equation in Eq. 2.37, except that the one in Eq. 2.37 has a univariate output, whereas the model above has a multivariate response.

Assume $\boldsymbol{\varepsilon} \sim \mathcal{N}(\mathbf{0}, \sigma_\varepsilon^2 \mathbf{I})$. Then, taking the variance of both sides of Eq. 12.2 and furthermore assuming independence between $\boldsymbol{x}$ and $\boldsymbol{\varepsilon}$, one has

$$\boldsymbol{\Sigma}_{\boldsymbol{y}} = \mathbf{H}\boldsymbol{\Sigma}_{\boldsymbol{x}}\mathbf{H}^T + \sigma_\varepsilon^2 \mathbf{I}. \tag{12.3}$$

Suppose that one of the elements in $\boldsymbol{x}$, say, x_i, is malfunctioning. As a result, x_i creates a substantially large variation source of the magnitude of σ_i^2. Assume that all other elements in $\boldsymbol{x}$ are properly functioning and thus have zero variance, or a variance so small relative to σ_i^2 that it can be approximated by zero. As such,

$$\boldsymbol{\Sigma}_{\boldsymbol{x}} = \begin{pmatrix} 0 & & & & \\ & \ddots & & & \\ & & \sigma_i^2 & & \\ & & & \ddots & \\ & & & & 0 \end{pmatrix}. \tag{12.4}$$

Substituting the above $\boldsymbol{\Sigma}_{\boldsymbol{x}}$ into Eq. 12.3, one gets

$$\boldsymbol{\Sigma}_{\boldsymbol{y}} = \sigma_i^2 \mathbf{h}_i \mathbf{h}_i^T + \sigma_\varepsilon^2 \mathbf{I}, \tag{12.5}$$

where $\mathbf{h}_i$ is the i-th column of $\mathbf{H}$.

With the presence of a systematic fault, the magnitude of the background noise, measured by σ_ε^2, is supposed to be much smaller than that of the fault, σ_i^2; otherwise, the fault may not be a real fault, or it may not be detectable. Aware of this, let us approximate Eq. 12.5 by dropping the term of the background noise. So the approximation reads

$$\boldsymbol{\Sigma}_{\boldsymbol{y}} \simeq \sigma_i^2 \mathbf{h}_i \mathbf{h}_i^T. \tag{12.6}$$

What Eq. 12.6 implies is that $\mathbf{h}_i$ is an eigenvector of $\boldsymbol{\Sigma}_{\boldsymbol{y}}$. To see this, applying $\boldsymbol{\Sigma}_{\boldsymbol{y}}$ to $\mathbf{h}_i$, one gets

$$\boldsymbol{\Sigma}_{\boldsymbol{y}}\mathbf{h}_i = \sigma_i^2 \mathbf{h}_i \mathbf{h}_i^T \mathbf{h}_i = \lambda_i \mathbf{h}_i, \tag{12.7}$$

where λ_i is the corresponding eigenvalue, taking the value of $\lambda_i = \sigma_i^2 \|\mathbf{h}_i\|_2^2$. Of course, when there exists background noise, the noise's presence may create some perturbation to the eigenvector pattern. In the special case of having an uncorrelated noise (so that the noise covariance matrix is of the form $\sigma_\varepsilon^2 \mathbf{I}$), the eigenvector pattern will not be affected; just that the magnitudes of the eigenvalues change (see Exercise 12.2).

The above analysis leads to the signature-based diagnosis procedure summarized in Algorithm 12.1. In an eigenvalue analysis, most commercial software produces the set of eigenvectors in Step 5 to be unit vectors. To facilitate the comparison in Step 7, it is a good idea to normalize the column vectors in $\mathbf{H}$ while creating the signature library.

Algorithm 12.1 Linear modeling and signature-based fault diagnosis.

1. Establish a linear model as in Eq. 12.2 for the engineering system of interest.
2. The library of the fault signatures can be formed by taking the column vectors of the system matrix $\mathbf{H}$. This modeling process is conducted offline and based on physical and engineering principles governing the said system.
3. During the online process, collect the data of the response, $\boldsymbol{y}$.
4. Calculate its sample covariance matrix $\mathbf{S}_{\boldsymbol{y}}$ and use it as the estimation of $\boldsymbol{\Sigma}_{\boldsymbol{y}}$.
5. Compute the eigenvalues and eigenvectors of $\mathbf{S}_{\boldsymbol{y}}$.
6. Locate the eigenvector corresponding to the largest eigenvalue. This eigenvector is the estimated fault signature.
7. Compare the eigenvector located in Step 6 with the column vectors in the signature library. A statistical test is usually necessary, due to the presence of background noise and the use of the sample covariance matrix $\mathbf{S}_{\boldsymbol{y}}$. Identify the fault source based on a signature matching criterion.

12.3 SIMILARITY METRICS

In both anomaly detection and fault diagnosis, a central question is how to define the similarity (or dissimilarity) between data instances. It is evident

that without a similarity metric, it is impossible to entertain the concept of anomaly, as being anomalous means different, and a data instance is an anomaly because it is so substantially different from the rest of instances in a group. The similarity metric is equally crucial in the mission of diagnosis. For the supervised learning-based approach, the similarity metric is embedded in the loss functions. For the signature-based approach, a similarity metric is used explicitly to decide the outcome in the matching and comparison step. We discuss in this section a few schools of thoughts concerning the similarity metrics.

12.3.1 Norm and Distance Metrics

In an n-dimensional vector space, $\mathcal{H}_{n\times 1}$, the length of a vector or the distance between two points is defined through the concept of norm, which is a function mapping from the vector space to the nonnegative half of the real axis, i.e.,

$$\mathcal{H}_{n\times 1} \longmapsto [0, +\infty).$$

Consider a vector $\boldsymbol{x}$. Its p-norm, $p \geq 1$, is defined as

$$\|\boldsymbol{x}\|_p := \left(|x_1|^p + |x_2|^p + \ldots + |x_n|^p\right)^{\frac{1}{p}}. \tag{12.8}$$

When $p = 2$, the above definition is the 2-norm, also known as the Euclidean distance, that we use repeatedly throughout the book. When $p = 1$, the above definition gives the 1-norm, also nicknamed the Manhattan distance. The definition of a p-norm is valid when $p = \infty$, known as the ∞-norm, defined as

$$\|\boldsymbol{x}\|_\infty := \max\{|x_1|, |x_2|, \ldots, |x_n|\}. \tag{12.9}$$

When $0 < p < 1$, the expression in Eq. 12.8 is no longer a norm, because the triangular inequality condition, required in the definition of a valid norm, is not satisfied. When $p = 0$, the expression in Eq. 12.8 is called the 0-norm, which is also not a valid norm. Nevertheless, analysts use the 0-norm as a convenient notation to denote the number of non-zero elements in a vector.

The norm, $\|\boldsymbol{x}\|_p$, is the length of vector $\boldsymbol{x}$ and can be considered as the distance between the point, $\boldsymbol{x}$, and the origin. For two points in the vector space, $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, the distance between them follows the same definition as in Eq. 12.8 or Eq. 12.9 after replacing $\boldsymbol{x}$ by $\boldsymbol{x}_i - \boldsymbol{x}_j$.

The p-norm has a nice geometric interpretation. Fig. 12.4 illustrates the boundaries defined by $\|\boldsymbol{x}\|_p = 1$ in a 2-dimensional space. The boundary of the 1-norm, $\|\boldsymbol{x}\|_1 = 1$, is the diamond shape, that of the 2-norm, $\|\boldsymbol{x}\|_2 = 1$, is the circle, and that of the ∞-norm, $\|\boldsymbol{x}\|_\infty = 1$, is the square. When $p > 2$, the boundary is a convex shape between the circle and the square. When $p < 1$, even though $\|\boldsymbol{x}\|_p$ is no longer a proper norm, the boundary of $\|\boldsymbol{x}\|_p = 1$ can be visualized on the same plot, as the concave shapes inside the diamond.

The 2-norm, corresponding to the Euclidean distance, is the shortest distance between two points in a Euclidean space. This 2-norm metric measures

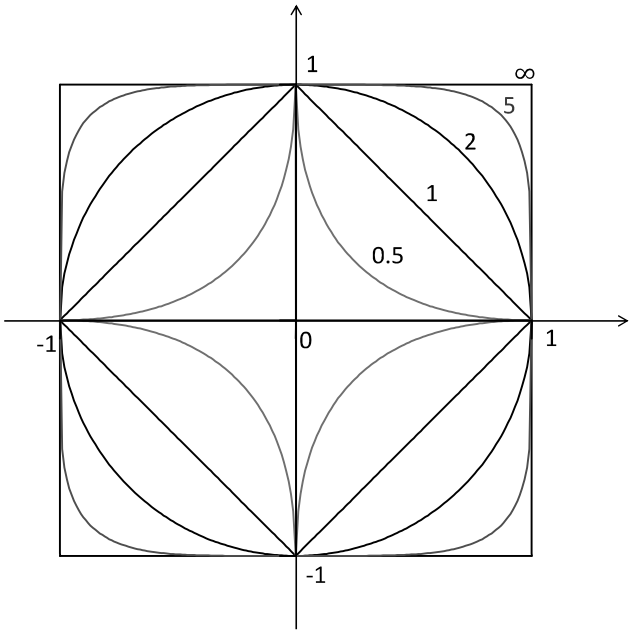


FIGURE 12.4 Boundaries defined by $\|\boldsymbol{x}\|_p = 1$.

the distance in our daily sense. It is widely used in many machine learning and data science methods as the metric defining the loss functions. It is arguably the most widely used similarity metric in anomaly detection.

12.3.2 Inner Product and Angle-Based Metrics

A similarity metric can also be defined as the angle between two vectors. This angle-based metric is particularly popular in the signature-based fault diagnosis.

To define the angle, the concept of inner product needs to be added to a vector space. In an n-dimensional vector space, $\mathcal{H}_{n\times 1}$, the inner product of two vectors, $\boldsymbol{x}$ and $\boldsymbol{y}$, is defined as

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle := \boldsymbol{x}^T \boldsymbol{y} = \sum_{i=1}^{n} x_i y_i, \tag{12.10}$$

where $\langle \cdot, \cdot \rangle$ is the notation used to denote an inner product. Given this definition, it is established that $\langle \boldsymbol{x}, \boldsymbol{x} \rangle = \|\boldsymbol{x}\|_2^2$.

In a Euclidean space, the angle, θ, formed by a pair of vectors can be defined by using the inner product, such that

$$\theta = \arccos\left(\frac{\langle \boldsymbol{x}, \boldsymbol{y} \rangle}{\|\boldsymbol{x}\|_2 \|\boldsymbol{y}\|_2} \right). \tag{12.11}$$

See Fig. 12.5, left panel, for an illustration.

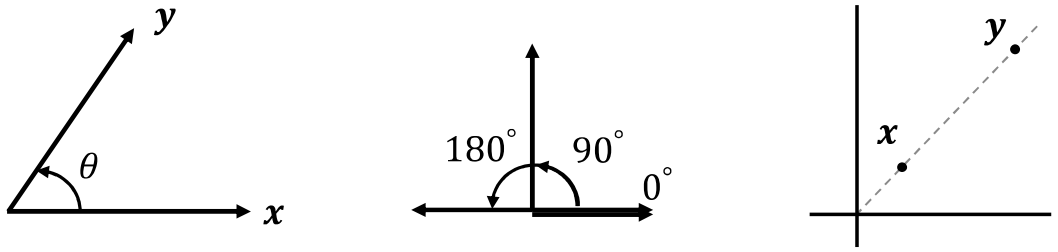


FIGURE 12.5 Angle-based similarity metric. Left panel: θ between two vectors; middle panel: three angular relationships; right panel: two data points on the same line in a vector space.

This angle, θ, can be used as a measure of similarity between two vectors. Take a look at Fig. 12.5, middle panel. If $\theta = 0°$, meaning that the two vectors are parallel and they point to the same direction, then these two vectors are considered the same, subject possibly to a difference in magnitude. If $\theta = 90°$, then the two vectors are said to be orthogonal to each other and they bear no similarity. If $\theta = 180°$, meaning that the two vectors are parallel but they point to the opposite directions, these vectors could still be considered the same, if the pointing direction does not matter in the context of an application. For this reason, some of the angle-based similarity criteria considers only the acute angles formed by two vectors.

The distance-based similarity metric and the angle-based similarity metric may serve different purposes in detection and diagnosis. The distance between two vectors depends on the lengths of them, but the angle does not. Look at the two data points in Fig. 12.5, right panel, which are on the same line but at different locations. The elements in $\boldsymbol{x}$ are proportional to those in $\boldsymbol{y}$, so that the angle between $\boldsymbol{x}$ and $\boldsymbol{y}$ is zero. If the two data points are considered different, then, the angle-based metric cannot signal such difference; the distance-based metric must be used instead. In some applications, such as in the signature-based diagnosis, however, what matters is the pattern exhibited by the relative magnitudes among the elements in a vector, rather than the absolute magnitudes. Recall that in the signature-based diagnosis, eigenvectors are normalized to be unit vectors, so that the vector lengths are neutralized. In that circumstance, the angle-based measure is a better metric. The distance-based metric can still be used if the vectors involved are normalized before comparison.

One advantage of using the angle-based similarity metric is its robustness in a high-dimensional space, as compared to the distance-based metric. When comparing two vectors, $\boldsymbol{x}$ and $\boldsymbol{y}$, in an n-dimensional space, the Euclidean distance, $\|\boldsymbol{x} - \boldsymbol{y}\|_2$, is affected more by the background noise embedded in the two vectors than the angle between them.

The distance-based and angle-based metrics can be connected. Recall the

kernel function $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ used in the formulation of SVM—revisit Eqs. 2.47 and 2.48. Note that the kernel function is the exponential of the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. Consider a reproducing kernel Hilbert space induced by the defined kernel function, $K(\cdot, \cdot)$. This RKHS is spanned by a possibly infinite set of basis functions, denoted as $\boldsymbol{\phi}(\boldsymbol{x}) = (\phi_1(\boldsymbol{x}), \phi_2(\boldsymbol{x}), \ldots, \phi_\ell(\boldsymbol{x}), \ldots)$. The theory of RKHS [86] tells us that

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \boldsymbol{\phi}(\boldsymbol{x}_i), \boldsymbol{\phi}(\boldsymbol{x}_j) \rangle,$$

which connects the distance-based metric in the left-hand side with the angle-based metric in the right-hand side. This above result underlies the well-known *kernel trick*. The RKHS basis functions on the right-hand side provides theoretical foundation for how an unknown function is reconstructed by learning through the training data. But the basis functions themselves are difficult to express analytically in closed forms. On the other hand, the kernel functions, such as the radial basis kernel in Eq. 2.48, can be easily expressed in simple, closed forms. With the equality above, one does not have to worry about the RKHS basis function, $\boldsymbol{\phi}(\cdot)$, but can simply use the corresponding kernel function, $K(\cdot, \cdot)$, instead. This substitute is the trick referred to as the kernel trick.

12.3.3 Statistical Distance

The Mahalanobis distance [140] used in Chapter 7 is also known as the statistical distance. A statistical distance is to measure the distance between a data point from a distribution or two data points in a vector space by accounting for the variance structure associated with the vector space.

Consider an $\boldsymbol{x}$ and a $\boldsymbol{y}$ from the multivariate normal distribution, $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. The statistical distance between them is defined as

$$\mathrm{MD}(\boldsymbol{x}, \boldsymbol{y}) := \sqrt{(\boldsymbol{x} - \boldsymbol{y})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{y})}. \tag{12.12}$$

This expression is equivalent to Eq. 7.3.

The statistical distance, $\mathrm{MD}(\boldsymbol{x}, \boldsymbol{\mu})$, measures the distance between the observation of $\boldsymbol{x}$ and the distribution, $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Its interpretation is that the likelihood of obtaining $\boldsymbol{x}$ as an observation from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ can be quantified by this statistical distance—the smaller the distance is, more likely that $\boldsymbol{x}$ will be observed (or the larger the likelihood of the observation).

The statistical distance between two samples is a weighted distance, whereas the Euclidean distance is an un-weighted distance. Given the same Euclidean distance between two points, their respective statistical distance could be different and is in fact re-scaled by the variance along the direction of the distance in question. Intuitively speaking, variance implies uncertainty. The vector space embodying the data are re-shaped by the level of uncertainty. Along the axis of low uncertainty, the scale is magnified (an old one mile could count as ten), whereas along the axis of high uncertainty, the scale is suppressed (an old ten miles may count only as one).

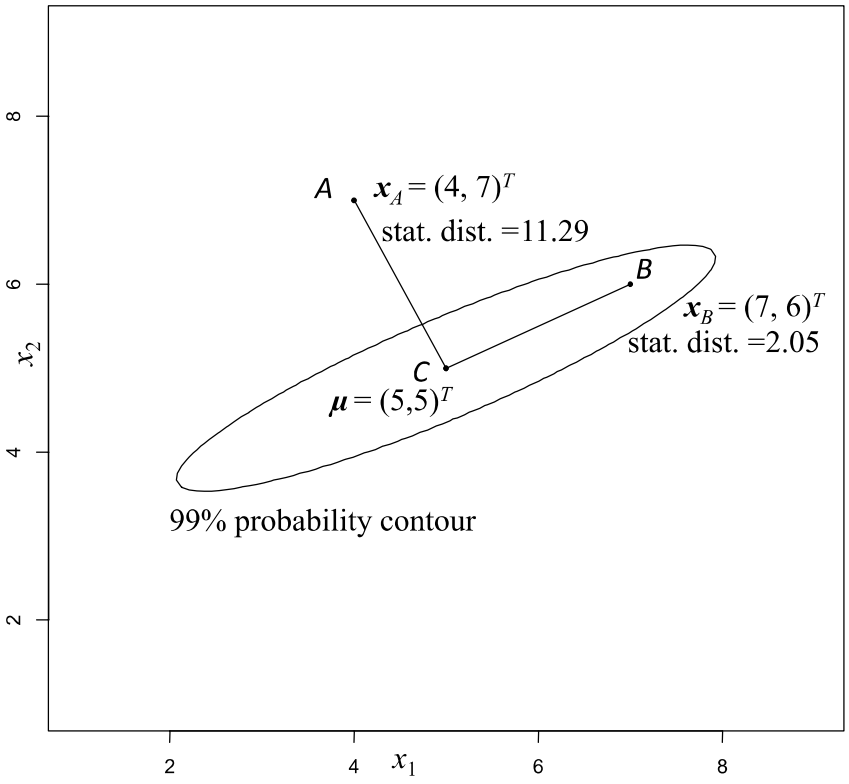


FIGURE 12.6 Statistical distance versus Euclidean distance. The elliptic contour is the 99% probability contour of a bivariate normal distribution.

As illustrated in Fig. 12.6, points A and B have the same Euclidean distance from the distribution center, C, but the respective statistical distances are very different. $\text{MD}(A, C)$ is a whole lot greater than $\text{MD}(B, C)$, because the vector, AC, aligns with the direction of a much smaller variance than the vector, BC. The distance between any point on the 99% probability contour and C is the same, although the respective Euclidean distance varies.

12.3.4 Geodesic Distance

When a vector space is unstructured, so that any pair of points in the space can reach each other in a straight line, that straight line is the shortest path between the pair of points, and the distance between them is measured by the corresponding Euclidean distance. But when a space is structured or curved, meaning that certain pathways are no longer possible, then, the shortest path between two points may not be a straight line anymore. One example is the shortest flight route between two cities on the surface of the earth. Because the flight route is constrained by the earth's surface and the shortest route

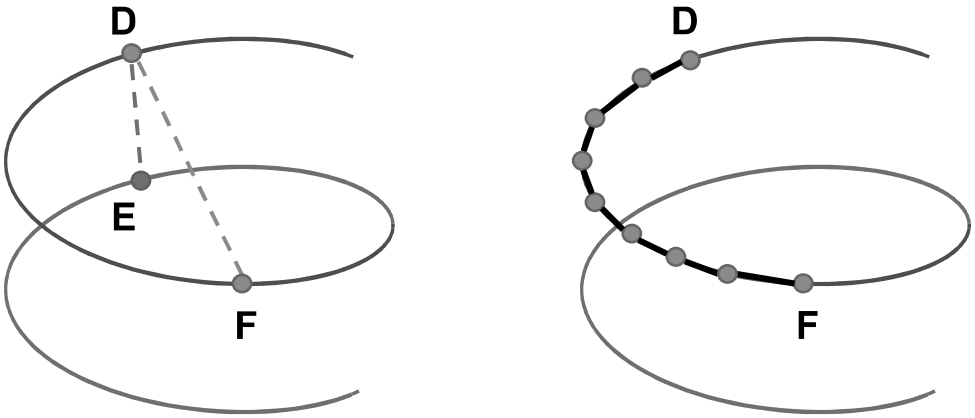


FIGURE 12.7 Left panel: geodesic distance versus Euclidean distance in a space with a swirl structure; right panel: the geodesic distance is approximated by the summation of a series of Euclidean distances associated with the short hops.

between two cities is a curved line, not a straight line. The distance in a structured space is measured by a geodesic distance.

Fig. 12.7 presents an example, inspired by the Swiss roll example presented by Tenenbaum et al. [213, Fig. 3]. Suppose that the data are constrained along the swirl structure in the space. Given this structural constraint, the distance between D and F is shorter than that between D and E, because along the curve and going from D, one reaches F first before reaching E. The distance between D and F along the curve, or that between D and E, is the geodesic distance between them. Should the space is unstructured, then the straight line linking D and E is shorter than that linking D and F. The implication is that using the pairwise Euclidean distance in all circumstances could mislead a data science algorithm to wrongly rank the similarity between data instances—in this case, using the Euclidean distance deems E more similar to D, in spite of the fact that the opposite is true. Tenenbaum et al. [213] raise the awareness of the existence of structures in data spaces and its impact on machine learning tasks.

Computing the geodesic distance can be complicated. This distance is usually approximated by the summation of the short hops using a series of intermediate points in between; see Fig. 12.7, right panel. The distance of any of the short hops is still calculated using a Euclidean distance. But the summation is no longer Euclidean. In Section 12.5, a minimum spanning tree (MST) is used to capture the structure of the underlying data space. The geodesic distance between any two points is approximated by the shortest path linking them through the MST.

12.4 DISTANCE-BASED METHODS

In this section, a few methods focusing on detecting local, pointwise anomalies are introduced.

12.4.1 Nearest Neighborhood-based Method

The concept of k-nearest neighborhood is explained in Section 5.3.1. Ramaswamy et al. [172] base their definition of anomaly on the distance of a point's k-th nearest neighbor.

Denote by $D^k(\boldsymbol{x}_i)$ the distance between the k-th nearest neighbor of $\boldsymbol{x}_i$ and $\boldsymbol{x}_i$ itself. Ramaswamy et al.'s method is to compute $D^k(\boldsymbol{x}_i)$ for every data point in a dataset, rank them from the largest to the smallest, and declare the first N_o data instances as anomalies. In this method, the neighborhood size, k, and the anomaly cut-off, N_o, are prescribed. The resulting method is referred to as the kNN anomaly detection.

Angiulli and Pizzuti [9] follow this kNN idea but argue that instead of using the distance between the k-th nearest neighbor and the target data point, one should use the summation of all distances from the most nearest neighbor to the k-th nearest neighbor. Angiulli and Pizzuti call this summation the weight of data point $\boldsymbol{x}_i$. Same as in the kNN anomaly detection, this weight is used to rank all data points and classify the top N_o points as anomalies. This revised nearest neighborhood-based method is referred to as kNNW, with the "W" implying "weight."

12.4.2 Local Outlier Factor

Breunig et al. [22] introduce a local outlier factor (LOF) method that makes use of the k-th nearest neighbor distance. First, for a given neighborhood size k, Breunig et al. introduce the concept of a reachability distance. Denoted by RD^k, the reachability distance of the data point, $\boldsymbol{x}_*$, with respect to an arbitrary point in the dataset, $\boldsymbol{x}_i$, is

$$\mathrm{RD}^k(\boldsymbol{x}_*, \boldsymbol{x}_i) = \max\{D^k(\boldsymbol{x}_i), D(\boldsymbol{x}_*, \boldsymbol{x}_i)\}, \tag{12.13}$$

where $D(\boldsymbol{x}_*, \boldsymbol{x}_i)$ is the Euclidean distance between the two data points. Basically, the reachability distance is a lower bound truncated distance. When the two points are too close, their reachability distance is no smaller than the k-th nearest neighbor distance, whereas when the two data points are far away enough, their reachability distance is the actual distance between them. In a sense, the concept of reachability distance is like putting a shield on $\boldsymbol{x}_i$. The point $\boldsymbol{x}_*$ can reach to $\boldsymbol{x}_i$ up to its k-th nearest neighbor but not nearer. Breunig et al. state that using the reachability distance reduces the statistical fluctuation of the actual distance and exerts a smoothing effect.

Next, Breunig et al. [22] want to quantify the reachability density of points in the neighborhood of $\boldsymbol{x}_*$. Points that have a lower reachability density than

its neighbors are deemed anomalies. Note that here the term "density" does not mean a probability density but rather the number of data points per unit volume.

A local reachability density (LRD) is defined as

$$\mathrm{LRD}(\boldsymbol{x}_*) = \frac{1}{\left(\sum_{\boldsymbol{x}_i \in\ \mathfrak{N}_k(\boldsymbol{x}_*)} \mathrm{RD}^k(\boldsymbol{x}_*, \boldsymbol{x}_i)\right) / |\mathfrak{N}_k(\boldsymbol{x}_*)|}, \tag{12.14}$$

where $\mathfrak{N}_k(\boldsymbol{x}_*)$ is the k-nearest neighborhood of $\boldsymbol{x}_*$ and $|\cdot|$ takes the cardinality of a set. LRD of $\boldsymbol{x}_*$ is the inverse of the average reachability distance using data points in the k-nearest neighborhood of the same point. When the average reachability distance is large, the density is low.

Finally, Breunig et al. [22] define their anomaly score as the ratio of the average local reachability density of $\boldsymbol{x}_*$'s k-nearest neighbors over the local rechability density of $\boldsymbol{x}_*$ and label it as LOF, such as

$$\mathrm{LOF}(\boldsymbol{x}_*) = \frac{\left(\sum_{\boldsymbol{x}_i \in\ \mathfrak{N}_k(\boldsymbol{x}_*)} \mathrm{LRD}(\boldsymbol{x}_i)\right) / |\mathfrak{N}_k(\boldsymbol{x}_*)|}{\mathrm{LRD}(\boldsymbol{x}_*)}. \tag{12.15}$$

The smaller the local density of $\boldsymbol{x}_*$, the higher its LOF, and more likely it is an anomaly. The LOF scores, once computed for all data points, are used to rank the data instances. The tope N_o instances are declared anomalies.

12.4.3 Connectivity-based Outlier Factor

Tang et al. [207] argue that the reachability density proposed by Breunig et al. [22] only considers the distances but does not consider the connectivity among neighborhood points. Yet, a low density does not always imply an anomaly. Rather, one should look at the degree of isolation of the said data point, which can be measured by the lack of connectivity. In other words, a data point that is less connected to other data points in a neighborhood is more likely an anomaly. Tang et al. state that "*isolation can imply low density, but the other direction is not always true.*"

Tang et al. [207] introduce a connectivity-based outlier factor (COF) score, which is in spirit similar to the LRD ratio used in Eq. 12.15, but the respective LRD is replaced with a connectivity-based distance metric.

Tang et al. [207] first define the distance between two non-empty sets, $\mathcal{X}$ and $\mathcal{Y}$, that are also disjoint, i.e., $\mathcal{X} \bigcap \mathcal{Y} = \emptyset$, such that

$$D(\mathcal{X}, \mathcal{Y}) = \min\{D(\boldsymbol{x}, \boldsymbol{y}) : \forall \boldsymbol{x} \in \mathcal{X} \text{ and } \boldsymbol{y} \in \mathcal{Y}\}. \tag{12.16}$$

Consider a target point, $\boldsymbol{x}_*$, to be evaluated. Tang et al. [207] iteratively build a k-nearest neighborhood for $\boldsymbol{x}_*$ and establish the sequence of connection. The procedure is outlined in Algorithm 12.2. In this algorithm, $\mathcal{G}_t$ records all the neighbor points and $\mathcal{E}_t$ records the local, pairwise connection steps linking $\boldsymbol{x}_*$ from the nearest point to the farthest point in the neighborhood. This

neighborhood, $\mathcal{G}_t$, is different from $\mathfrak{N}_k(\boldsymbol{x}_*)$ in principle, as $\mathfrak{N}_k(\boldsymbol{x}_*)$ is decided based purely on pairwise distances without considering the sequence of connection. This sequence of connection information is what Tang et al. argue makes all the difference between COF and LOF.

Algorithm 12.2 Build the locally connected k-nearest neighborhood for $\boldsymbol{x}_*$. Let $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1}^N$ be the original dataset and k be the prescribed neighborhood size.

---

1. Let $t = 1$, $\mathcal{G}_t = \{\boldsymbol{x}_*\}$, $\boldsymbol{x}_{(0)} = \boldsymbol{x}_*$, $\mathcal{D}_t = \mathcal{D} \backslash \boldsymbol{x}_{(0)}$, and $\mathcal{E}_t = \emptyset$.
2. Find $\boldsymbol{x}_{(t)} \in \mathcal{D}_t$, such that $D(\mathcal{G}_t, \boldsymbol{x}_{(t)})$ is minimized.
3. Augment $\mathcal{G}_t$ such that $\mathcal{G}_t = \mathcal{G}_t \cup \{\boldsymbol{x}_{(t)}\}$.
4. Let $\mathcal{E}_t = \mathcal{E}_t \cup \{(\boldsymbol{x}_{(t-1)}, \boldsymbol{x}_{(t)})\}$.
5. Let $\mathcal{D}_t = \mathcal{D}_t \backslash \boldsymbol{x}_{(t)}$.
6. Let $t = t + 1$, and repeat from Step 2 until there are $k + 1$ elements in $\mathcal{G}_t$ (or k elements besides $\boldsymbol{x}_*$).

Once the neighborhood and its connectivity are established, Tang et al. [207] introduce the following connectivity-based distance, or as they call it, the *chaining distance*, denoted by $\mathrm{CD}_{\mathcal{G}}(\boldsymbol{x}_*)$, such that

$$\mathrm{CD}_{\mathcal{G}}(\boldsymbol{x}_*) = \frac{1}{k} \sum_{i=1}^{k} \frac{2(k+1-i)}{k+1} D(\boldsymbol{x}_{(i-1)}, \boldsymbol{x}_{(i)}), \tag{12.17}$$

where $(\boldsymbol{x}_{(i-1)}, \boldsymbol{x}_{(i)})$ is the i-th element in $\mathcal{E}$. Obviously, CD above is a weighted average distance, with a higher weight given to the connections closer to $\boldsymbol{x}_*$ and a lower weight given to the connections farther away from $\boldsymbol{x}_*$. Tang et al. choose the weight such that when $D(\boldsymbol{x}_{(i-1)}, \boldsymbol{x}_{(i)})$ is the same for all i's, the weight coefficients are summed to one (see Exercise 12.9).

Tang et al. [207] define their COF score, under a given k, as

$$\mathrm{COF}(\boldsymbol{x}_*) = \frac{\mathrm{CD}_{\mathcal{G}}(\boldsymbol{x}_*)}{\sum_{\boldsymbol{x}_i \in \mathcal{G}(\boldsymbol{x}_*)} \mathrm{CD}_{\mathcal{G}}(\boldsymbol{x}_i)/|\mathcal{G}(\boldsymbol{x}_*)|}. \tag{12.18}$$

The use of COF follows that of LOF. The larger a COF, the more likely the corresponding data instance is deemed an anomaly.

12.4.4 Subspace Outlying Degree

To deal with high-dimensional data problems, analysts choose to consider a subset of the original features, an action commonly known as *dimension reduction*. The potential benefit of looking into a subspace is that data points

distributed indistinguishably in the full dimensional space could deviate significantly from others when examined in a proper subspace. On the other hand, the danger of using a subspace approach is that if not chosen properly, the difference between a potential anomaly and normal points may disappear altogether in the subspace. It is obvious that the tricky part of a subspace approach is how to find the right subspace.

Kriegel et al. [125] present a subspace outlying degree (SOD) method. The method works as follows. First, Kriegel et al. compute the variance of the set of the reference points in $\mathcal{D}$ as

$$\text{VAR} = \frac{1}{|\mathcal{D}|} \sum_{\boldsymbol{x}_* \in \mathcal{D}} D(\boldsymbol{x}_*, \boldsymbol{\mu})^2, \tag{12.19}$$

where $\boldsymbol{\mu}$ is the average position of the points in $\mathcal{D}$. Similarly, compute the variance along the i-th attribute as

$$\text{VAR}_i = \frac{1}{|\mathcal{D}|} \sum_{\boldsymbol{x}_* \in \mathcal{D}} D((\boldsymbol{x}_*)_i, \boldsymbol{\mu}_i)^2, \tag{12.20}$$

where $(\boldsymbol{x}_*)_i$ and $\boldsymbol{\mu}_i$ are, respectively, the i-th element in $\boldsymbol{x}_*$ and $\boldsymbol{\mu}$.

Then, create a subspace vector based on the following criterion, where n is the dimension of the original data space and α is a constant,

$$\nu_i = \begin{cases} 1, & \text{if } \text{VAR}_i < \alpha \cdot \frac{\text{VAR}}{n}, \\ 0, & \text{otherwise.} \end{cases} \tag{12.21}$$

Kriegel et al. [125] suggest setting $\alpha = 0.8$. When ν_i in Eq. 12.21 is one, the corresponding variable is selected to construct the subspace; otherwise, the corresponding variable is skipped over. Denote the resulting subspace by $\mathcal{S}$, which is represented by the vector $\boldsymbol{\nu} = (\nu_1, \nu_2, \ldots, \nu_n)$. In a three-dimensional space, for instance, $\boldsymbol{\nu} = (1, 0, 1)$ indicates that the selected subspace is spanned by the first and third axes.

To measure the deviation of a data point, $\boldsymbol{x}_*$, from a subspace, Kriegel et al. [125] use the following formula,

$$D(\boldsymbol{x}_*, \mathcal{S}) := \sqrt{\sum_{i=1}^{n} \nu_i ((\boldsymbol{x}_*)_i - \boldsymbol{\mu}_i)^2}. \tag{12.22}$$

Kriegel et al. further define their SOD score as

$$\text{SOD}(\boldsymbol{x}_*) := \frac{D(\boldsymbol{x}_*, \mathcal{S})}{\|\boldsymbol{\nu}\|_1}, \tag{12.23}$$

where $\|\boldsymbol{\nu}\|_1$ is the number of dimensions of the selected subspace. A higher SOD score means that $\boldsymbol{x}_*$ deviates from the selected subspace a lot and is thus likely an anomaly.

12.5 GEODESIC DISTANCE-BASED METHOD

As explained in Section 12.3.4, the Euclidean-based similarity metric works well in an unstructured data space but could mislead a learning method when there are intrinsic structures in a data space restricting certain pathways connecting data points. In the circumstances of structured data spaces, a geodesic distance ought to be used. The methods introduced in Section 12.4 rely heavily on the use of Euclidean distance to define similarity, with the exception of COF. COF, through the use of the connection sequence, bears certain characteristics of the geodesic distance. In the benchmark case study of Section 12.6.1, COF does perform rather competitively. More recently, Ahmed et al. [4] develop an MST-based unsupervised anomaly detection method that takes full advantage of a geodesic distance-based similarity metric.

12.5.1 Graph Model of Data

The MST-based anomaly detection method employs a minimum spanning tree to approximate the geodesic distances between data points in a structured space and then uses the distance approximation as the similarity metric. The data are modeled as a network of nodes through a graph. Consider a connected undirected graph $G = (U, E)$, where U denotes the collection of vertices or nodes and E represents the collection of edges connecting these nodes as pairs. For each edge $e \in E$, there is a weight associated with it. It could be either the distance between the chosen pair of nodes or the cost to connect them.

A minimum spanning tree is a subset of the edges in E that connects all the nodes together, without any cycles and with the minimum possible total edge weight. This total edge weight, also known as the total length or total cost of the MST, is the summation of the weights of the individual edges. If one uses the Euclidean distance between a pair of nodes as the edge weight, the resulting spanning tree is called a Euclidean MST.

Consider the example in Fig. 12.8, where U=$\{1, 2, 3, 4\}$ and E =$\{e_{12}, e_{13}, e_{14}, e_{23}, e_{24}, e_{34}\}$. All edges in E are all different in length and the edge length order is specified in the left panel of Fig. 12.8. If one wants to connect all the nodes in U without forming a cycle, there could be 16 such combinations with only one having the minimum total edge length. That one is the MST for this connected graph, shown in the right panel of Fig. 12.8. Note that some of the edges in Fig. 12.8 look like having the same length. The edge e_{13} looks even longer than e_{34} and e_{23}. One way to imagine a layout satisfying the edge length order specified in Fig. 12.8 is to envision node #3 not in the same plane formed by node #1, #2, and #4, but hovering in the space and being close to node #1.

Ahmed et al. [4] consider data instances as nodes and the Euclidean distance between any pairs of data points as the edge weight and then construct an MST to connect all the nodes. Specifically, they use the algorithm in [168] to construct an MST. Although the distance between a pair of immediately

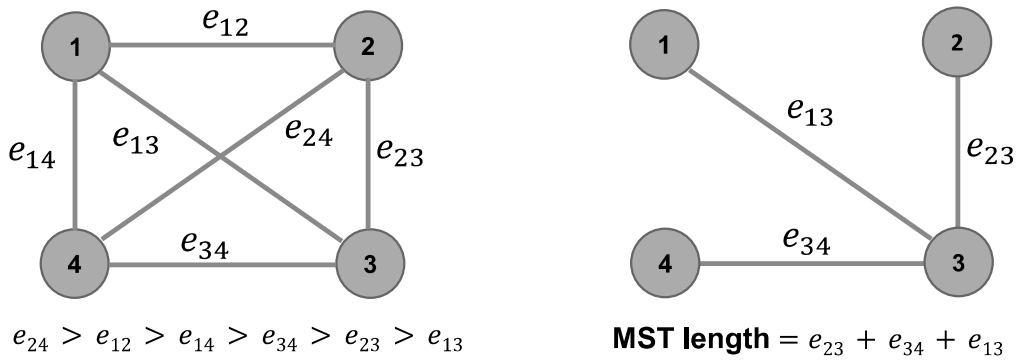


FIGURE 12.8 Formation of a minimum spanning tree. Left panel: the complete graph; right panel: the minimum spanning tree. (Reprinted with permission from Ahmed et al. [4].)

connected nodes is still Euclidean, the distance between a general pair of nodes (i.e., data points) is not. Rather, it is the summation of many small-step, localized Euclidean distances hopping from one data point to another point. The MST reflects the connectedness among data points in a structured space and the MST-based distance approximates the geodesic distance between two data points.

12.5.2 MST Score

Ahmed et al.'s method [4] focuses on detecting the local, pointwise anomalies. But the MST nature enables the method to incorporate a preprocessing step that can identify potential anomalous clusters. The idea is simple. First, build a global MST using all the data points. After the formation of the global MST, one can look for an unusual long edge and deem it as the connecting edge between an anomalous cluster and the rest of the MST. Once the long edge is disconnected, it separates the MST into two groups, and the smaller group is considered an anomalous cluster. The "unusual" aspect can be verified through a statistical test, say, longer than the 99th percentile of all edge lengths in the original MST. This preprocessing step can be iteratively applied to the remaining larger group, until no more splitting.

For detecting the local anomalies, one needs to go into the neighborhood level. Same as the local anomaly detection methods introduced in Section 12.4, two parameters are prescribed for the MST-based approach: the neighborhood size, k, and the cut-off threshold, N_o, for declaring anomalies.

Denote by $\mathcal{D}$ the set of data points to be examined, where $\mathcal{D}$ could be the whole original dataset or could be the remaining set of data after the anomalous cluster is removed. For a given data point in $\mathcal{D}$, first isolate its k nearest neighbors and treat them as this data point's neighborhood. Then, build an MST in this neighborhood. The localized, neighborhood-based MSTs

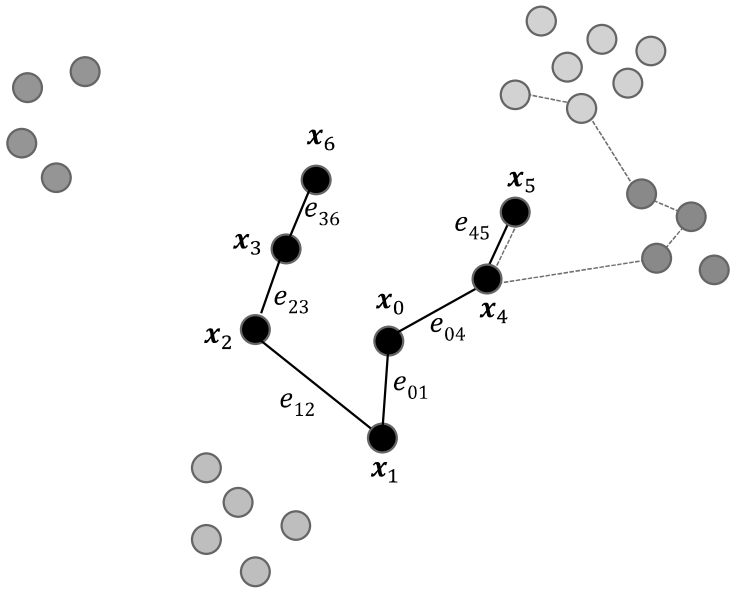


FIGURE 12.9 Local MST and LoMST score. The total edge weight of the local MST for $\boldsymbol{x}_0$ is its LoMST score, i.e., $W_{\boldsymbol{x}_0} = e_{01} + e_{12} + e_{23} + e_{36} + e_{04} + e_{45}$. (Reprinted with permission from Ahmed et al. [4].)

are referred to as *local MSTs* (LoMST). The total edge length of the LoMST associated a data point is called the LoMST score for this data point and is considered the metric measuring its connectedness with the rest of the points in the neighborhood as well as how far away it is from its neighbors. The LoMST is then used as the differentiating metric to signal the possibility that the said data point may be an anomaly.

Consider the illustrating example in Fig. 12.9. Suppose that one has chosen $k = 6$ and start with data point $\boldsymbol{x}_0$. Then, one can locate its neighbors as $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, $\boldsymbol{x}_3$, $\boldsymbol{x}_4$, $\boldsymbol{x}_5$ and $\boldsymbol{x}_6$. The MST construction algorithm connects $\boldsymbol{x}_0$ to its neighbors in the way as shown in Fig. 12.9. For $\boldsymbol{x}_0$, the total edge weight is $W_{\boldsymbol{x}_0} = e_{01} + e_{12} + e_{23} + e_{36} + e_{04} + e_{45}$, which is used as the LoMST score for $\boldsymbol{x}_0$. This procedure will be repeated for other data points. Fig. 12.9 does show another MST, which is for $\boldsymbol{x}_5$ in the dotted edges.

The LoMST score for a selected data instance is compared with its neighbor's score. The steps of comparison are to be repeated to cover all nodes in $\mathcal{D}$. Then the comparison scores are normalized to be between zero and one. The resulting normalized scores are also referred to as the LoMST scores, as long as there is no ambiguity in the context. The normalized LoMST scores are sorted in decreasing order, so that the top N_o instances are flagged as anomalies. The method is summarized in Algorithm 12.3.

Algorithm 12.3 MST-based anomaly detection method. Input: dataset $\mathcal{D}$, rows represent observations and columns represent attributes, the neighborhood size, k, and the cut-off level for identifying anomalies, N_o. Output: the anomaly index set, $\widehat{\mathcal{O}}$.

1. Preprocess to remove obvious anomalous clusters, if necessary.
2. Set $\mathcal{T} = \emptyset$, $\widehat{\mathcal{O}} = \emptyset$, $i = 1$.
3. For $\boldsymbol{x}_i \in \mathcal{D}$, determine its k nearest neighbors and save them in U_i.
4. Construct a complete graph using nodes in U_i. The resulting edges are in the set, E_i.
5. Construct a local MST using the edges in E_i.
6. Calculate the total length of $\boldsymbol{x}_i$'s local MST and denote it as $W_{\boldsymbol{x}_i}$.
7. Calculate the average of the total length of the LoMSTs associated with all nodes in U_i, and denote the average as $\overline{W}_i$.
8. Calculate the LoMST score for $\boldsymbol{x}_i$ as $T_i = W_{\boldsymbol{x}_i} - \overline{W}_i$.
9. Let $\mathcal{T} = \mathcal{T} \cup \{T_i\}$ and $i = i + 1$. Re-iterate from Step 3 until all data points in $\mathcal{D}$ are visited.
10. Normalize the scores stored in $\mathcal{T}$ to be between 0 and 1.
11. Rank the normalized scores in $\mathcal{T}$ in descending order.
12. Identify the top N_o scores and store the corresponding observations as point anomalies in $\widehat{\mathcal{O}}$.

12.5.3 Determine Neighborhood Size

For the neighborhood-based methods, including LoMST, an important parameter to be specified prior to the execution of a respective algorithm is the neighborhood size k. The difficulty in choosing k in an unsupervised setting is that methods like cross validation that work for supervised learning do not apply here. Ahmed et al. [4] advocate an approach based on the following observations, illustrated in Fig. 12.10 using two benchmark datasets.

When Ahmed et al. [4] plot the average LoMST scores for a broad range of k (here 1–100), they observe that at small k values, the average LoMST score tends to fluctuate, but as they keep increasing k, the average LoMST score tends to become stable at certain point. This leads to the understanding that when a proper k is chosen and the structure of the data is revealed, the label of the instances become fixed; such stability is reflected in a less fluctuating LoMST score. If one keeps increasing k, there is the possibility that the data structure becomes mismatch with the assigned number of clusters, so that the current assignments of anomalies and normal instances become destabilized. Consequently, the average LoMST score could fluctuate again. Based on this observation, a sensible strategy in choosing k is to select a range of k where the average LoMST scores are stable. If there are more than one stable ranges, analysts are advised to select the first one.

Let us look at the examples in Fig. 12.10. For the `Cardiotocography` dataset, Ahmed et al. [4] choose a k ranging from 27–47 and for the `Glass` dataset, they choose a k ranging from 70–95. Within the identified stable range, which k to choose matters less. What Ahmed et al. suggest is to select the k value that returns the maximum standard deviation of the LoMST scores, because by maximizing the standard deviation among the LoMST scores, it increases the separation between the normal instances and anomalous instances and facilitates the detection mission.

12.6 CASE STUDY

12.6.1 Benchmark Cases

As mentioned early in this chapter, one profound difficulty in assessing the performance of anomaly detection method is due to the lack of knowledge of the ground truth. Luckily, Campos et al. [30] published a comprehensive survey on the topic of anomaly detection and collected and shared 20 benchmark datasets of wide varieties, for each of which the anomalies are known. Readers are directed to the following website to retrieve the datasets, i.e., `http://www.dbs.ifi.lmu.de/research/outlier-evaluation/`, that hosts the supplemental material and online repository of [30]. Several versions of these datasets are available. These versions mainly differ in terms of the preprocessing steps used. What is used in this section is the normalized version of the datasets in which all the missing values are removed and categorical variables are converted into numerical format.

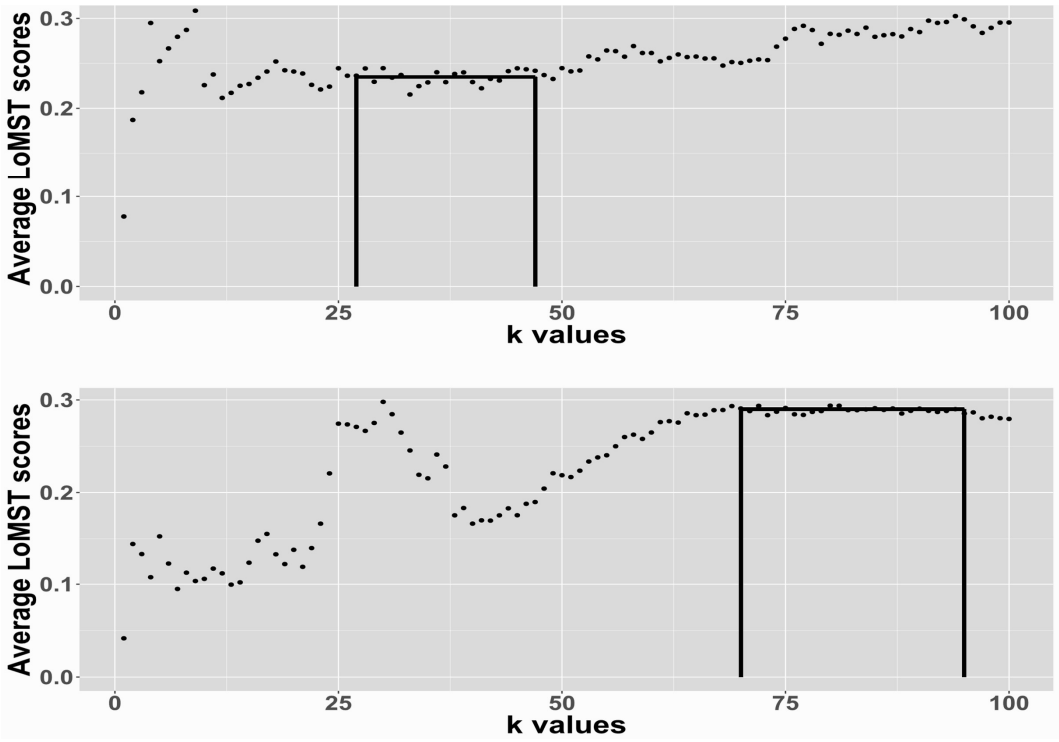


FIGURE 12.10 Selection of the neighborhood size k. Top panel: the `Cardiotocography` dataset; bottom panel: the `Glass` dataset. (Reprinted with permission from Ahmed et al. [4].)

TABLE 12.1 Benchmark datasets and the performance of LoMST under different data-to-attribute ratios.

Dataset	N/n	N	n	$\lvert\mathcal{O}\rvert$	Rank (Best k)	Rank (Practical k)
Arrhythmia	2	450	259	12	1	1
SpamBase	81	4,601	57	280	1	1
KDDcup99	1,479	60,632	41	200	3	2
WPBC	6	198	33	47	1	3
Ionosphere	11	351	32	126	2	3
WBC	51	367	30	10	1	1
ALOI	1,852	50,000	27	1,508	7	7
Parkinson	9	195	22	5	1	1
Annthyroid	343	7,200	21	347	11	10
Waveform	164	3,443	21	100	1	1
Cardiotocography	102	2,126	21	86	2	3
Lymphography	8	148	19	6	1	1
Pendigits	617	9,868	16	20	1	2
HeartDisease	21	270	13	7	1	1
PageBlocks	548	5,473	10	99	4	7
Stamps	38	340	9	16	1	6
Shuttle	113	1,013	9	13	1	1
WDBC	13	454	9	10	2	2
Pima	96	768	8	26	1	1
Glass	31	214	7	9	1	1

Source: Ahmed et al. [4]. With permission.

Table 12.1 summarizes the basic characteristics of the 20 benchmark datasets, for each of which N is the total number of data amount, n is the number of attributes, and $|\mathcal{O}|$ is the number of anomalies. Table 12.1 also presents the data-to-attribute ratio, N/n, which is rounded to the nearest integer. The two columns under the "Rank" headings in Table 12.1 are to be explained later.

Ahmed et al. [4] focus on the neighborhood-based approaches. These approaches include LoMST, a few others introduced in Section 12.4, and additional approaches that are not explained in this chapter but their details can be found in [30]. All these methods have a common parameter, which is the neighborhood size, k. Since the SPC method can also be used for the purpose of anomaly detection, it is included in the study as well. The specific SPC technique used is the Hotelling T^2 control chart. In total, 14 competitive methods are evaluated in this benchmark case study. The unexplained acronyms used in the subsequent tables are: local density factor (LDF), outlier detection using indegree number (ODIN), simplified local outlier factor (SLOF), local outlier probabilities (LoOP), influenced outlierness (INFLO), local distance-based outlier factor (LDOF), fast angle-based outlier detection (FABOD), and kernel density estimation outlier score (KDEOS).

TABLE 12.2 Performance comparison based on the best k value.

	LoMST	COF	LDF	kNN	ODIN	LOF	kNNW
Better	6	0	3	1	1	0	1
Equal	7	5	5	2	1	3	2
Close	5	10	6	7	8	8	7
Worse	2	5	6	10	10	9	10
Rank	2.2	3.3	3.8	5.0	7.7	5.1	4.5

	SLOF	LoOP	INFLO	LDOF	FABOD	KDEOS	SPC
Better	0	0	0	0	0	0	1
Equal	2	2	2	1	2	2	0
Close	6	6	5	7	5	1	1
Worse	12	12	13	12	13	17	18
Rank	5.9	5.8	6.2	7.9	7.0	8.9	11.7

Source: Ahmed et al. [4]. With permission.

Campos et al. [30] do not specify how to select k. They try a range of k values (from 1 to 100) to obtain all the results and then choose the best k value for each method. In the first comparison, Ahmed et al. [4] follow the same approach and label this as the "best k" comparison. The performance criterion used here is the precision at N_o, $P@N_o$, explained in Section 12.1.3. The results are presented in Table 12.2. Please note that the best k value in Table 12.2 may be different for respective methods. To better reflect the detection capability as they are compared to one another, Ahmed et al. break down the comparative performance into four major categories, namely *Better*, *Equal*, *Close* and *Worse*. Their meanings are as follows:

- *Better*, if a method is uniquely the best among all candidates.
- *Equal*, if a method ties with other methods for being the best.
- *Close*, if a method's correct detections are within 20% of the best alternative(s).
- *Worse*, if a method's correct detections are more than 20% lower than that of the best alternative(s).

If analysts rank each of the 14 methods in a scale of 1 to 14 according to its actual performance in relative to others, then an average relative rank can be calculated for each method. The average ranks are reported in the row below the four categories—the smaller, the better. The LoMST method's average rank is 2.2, with some closest competitors being COF (3.3), LDF (3.8), kNNW (4.5), kNN (5.0) and LOF (5.1).

Understandably, the "best k" is not practical, as analysts in reality do not know the anomalies while selecting k. Using the strategy devised in Section 12.5.3 to select a practical value of k for LoMST, Ahmed et al. [4] apply

TABLE 12.3 Performance comparison based on the practical k value.

	LoMST	COF	LDF	kNN	ODIN	LOF	kNNW
Better	5	2	1	1	2	0	0
Equal	5	1	4	5	1	3	4
Close	7	11	4	6	8	7	9
Worse	3	6	11	8	9	10	7
Rank	2.8	4.2	5.7	5.3	6.7	6.1	4.3

	SLOF	LoOP	INFLO	LDOF	FABOD	KDEOS	SPC
Better	0	0	0	0	2	0	1
Equal	1	1	2	1	2	0	0
Close	8	7	7	6	6	3	3
Worse	11	12	11	13	10	17	16
Rank	6.5	5.6	5.8	7.6	4.9	11.7	8.7

Source: Ahmed et al. [4]. With permission.

the same k to the other 12 alternative methods that need this value (SPC does not need to know k). The performance comparison based on the practical k is presented in Table 12.3, arranged in the same way as Table 12.2. Under the "practical k," the average rank of LoMST is 2.8, with some closest competitors being COF(4.2), kNNW (4.3), FABOD(4.9), kNN (5.3), and LDF (5.7).

Ahmed et al. [4] summarize LoMST's performance with respect to the data size and report the values under the last two columns in Table 12.1. Looking at the two extremes, the case of the highest number of observations ($N = 60,632$, `KDDcup99`), which is the one having the second highest N/n ratio, versus the case of the highest number of attributes ($n = 259$, `Arrhythmia`), which is also the one having the lowest N/n ratio, LoMST performs on top in both cases. It can also be noticed that on two of the datasets when the number of anomalies are too numerous (over a few hundreds to more than one thousand), LoMST does not do well enough. In hindsight, it makes intuitive sense, as LoMST is designed to find the local, pointwise anomalies, which, when existing, should be of a relatively small amount.

Another note is about the computational complexity of LoMST, which comes from two major sources. First, one needs to conduct the k-nearest neighbor search based on the chosen k. Then, for each observation, one needs to build a local MST using its k-nearest neighbors. For the first step, Ahmed et al. [4] use the fast approximate nearest neighbor searching approach [10, 16] with a time complexity of $O(nN \log N) + O(kn \log N)$. The first time complexity component, $O(nN \log N)$, represents the time to build the tree structure, whereas the second component, $O(kn \log N)$, represents the k-nearest neighborhood query time for a single observation. In the second step, building the local MST has the time complexity of $O(|U| \log |E|)$. Because LoMST is a localized, neighborhood-based MST, the values of $|U|$ and $|E|$ depend on k but usually remain small. The neighborhood search and the local MST step

are repeated N times, whereas the tree structure building is a one-time action. As such, the total complexity of the LoMST algorithm is approximately $\mathrm{O}(nN \log N) + \mathrm{O}(N[kn \log N + |U| \log |E|])$.

12.6.2 Hydropower Plant Case

The hydropower data initially received are time-stamped for a duration of seven months. The data are collected from different functional areas in the plant, such as the turbines, generators, bearings, and so on. The data records are collected at 10-minute intervals. Missing data are common. There are a total of 9,508 observations (rows in a data table) and 222 attribute variables (columns in a data table). Each row has a time stamp assigned to it. Attribute variables are primarily temperatures, vibrations, pressure, harmonic values, active power, etc. Before applying the anomaly detection method, some basic preprocessing and statistical analysis are performed in order to clean the data. To maintain the similarity with the 20 benchmark datasets, the hydropower data are normalized. After the preprocessing, the total number of observations comes down to 9,219. In summary, for the hydropower plant dataset, $N = 9,219$, $n = 222$, and $|\mathcal{O}|$ is unknown. The details of data preprocessing can be found in [4].

Besides LoMST, Ahmed et al. [4] apply two other popular anomaly detection methods to the same hydropower data. The two other methods are LOF [22] and SOD [125]. Here, LOF is used as a representative of the neighborhood-based methods, while SOD is a representative of the subspace methods. Note that SOD is not included in the benchmark case study, because the methods included in Section 12.6.1 are primarily neighborhood based. Had SOD been included in the benchmark study, under the practical k setting, SOD would have an average rank of 6.7 and the number of instances of its detection in the four categories would be 0, 0, 8, and 12, respectively.

For all three methods, one needs to specify the value of the nearest neighbors k. In this case, it would be great to get some suggestions from the domain experts about the possible size of an anomaly cluster based on their knowledge of the system. Ahmed et al. [4] indeed receive advice from their industrial collaborators, consider the value of k in a range of 10–20, and find the anomaly scores for each k in the range. Then they take the average of the resulting anomaly scores as the final anomaly score for each of the data instances. For SOD, one needs to select two parameters instead of one—one parameter is k, while the other one is the number of reference points for forming the subspace. To maintain the comparability with LoMST and LOF, Ahmed et al. choose $k = 15$ for SOD, which is the middle point of the above-suggested range. Concerning the number of reference points, it should be smaller than k but not too small, because an overly small number of reference points may render instability in SOD. Ahmed et al. explore a few options and finally settle on ten. Below ten, the SOD method becomes unstable.

By applying the three methods, the top 100 anomalies identified are shown

in Table 12.4. It can be observed that after the top 30 time stamps, no new anomaly-prone days emerge. Rather similar data patterns keep repeating themselves with slight differences in the time stamps. We therefore skip some rows after the top 30 stamps in Table 12.4.

The performance of the three methods are reasonably consistent, as 14 out of the top 30 probable anomalies identified by these methods are common, represented by an asterisk (*) in Table 12.4. This similarity continues even if one goes beyond the top 30 time stamps. By looking closely at these top 100 time stamps, one may find that there are some particular days and certain time chunks on these days which are more prone to anomaly. Such a pattern makes sense, as for most of engineering systems, random anomalies happen sparsely, while systematic anomalies take place in a cluster.

These three methods work differently, especially since SOD is from another family of methods. In spite of their differences, they have returned similar results for the hydropower dataset. This serves as a way to cross validate the detection outcomes while the true anomalies are unknown.

The three methods do have differences in their detection outcomes. The LOF method completely misses the 4th of July time stamps, although almost half of the 100 top anomaly-prone time stamps returned by both SOD and LoMST methods belong to that day. Ahmed et al. [4] investigate the issue and find that most of the time stamps in July correspond to low active power, whereas the time stamps from the 4th of July are marked with abnormally high active power. The rest of the attributes behave almost identically as other days of July. When the number of attributes increases, the nearest neighborhood methods usually fall short of detecting anomalies if abnormal values only happen to one or a few dimensions. This is where the subspaces method can do better (assuming that the abnormal value subspace is successfully identified). It is therefore not surprising to see that SOD detects these anomalies correctly. It is also encouraging to see that LoMST is capable of detecting these anomalies as well, even though LoMST is a neighborhood-based method. On the other hand, LOF and LoMST, being local methods, successfully identified point anomalies on the 16th of April, whereas the SOD method fails to identify them. In a nutshell, the LoMST method attains the merit of subspace-based methods without losing the benefits of local neighborhood-based methods.

Anomaly detection does not immediately reveal the root causes of the anomalies. Finding out which variables contribute to the anomalies helps the domain expert to verify the root causes and then fix them. This calls for a diagnostic follow-up. As presented in Section 12.2.1, after the anomalies are identified, the original unsupervised learning problem is translated into a supervised learning problem. While doing so, Ahmed et al. [4] discard the July 4th time stamps from the top 100 time stamps, as the reason for that happening is straightforward. They proceed with the remaining of the top 100 time stamps and assign them a response value of one (meaning an anomaly), and all other data records outside the top 100 time stamps a response value of zero (meaning a normal condition). Ahmed et al. then build a CART using the

TABLE 12.4 Summary of the top 100 anomalies in the hydropower dataset. Events followed by asterisk (*) are the common ones identified by all three methods in the top 30 time stamps.

LoMST	LOF	SOD
1/12/2016 11:20*	1/12/2016 11:30*	9/14/2015 8:00
9/14/2015 1:00*	9/14/2015 1:00*	1/12/2016 11:30*
1/2/2016 9:10*	9/14/2015 1:10*	9/13/2015 7:00*
1/11/2016 12:00*	1/12/2016 11:20*	7/4/2015 8:30
1/2/2016 9:30*	1/9/2016 6:50*	7/4/2015 8:20
7/4/2015 11:20	1/2/2016 9:10*	9/14/2015 1:50
7/4/2015 11:10	9/14/2015 8:00	7/4/2015 5:40
7/4/2015 11:30	1/2/2016 9:20*	1/11/2016 12:00*
1/9/2016 6:50*	1/9/2016 18:30	9/14/2015 1:00*
7/4/2015 10:40	9/14/2015 8:10*	10/3/2015 14:40
1/2/2016 9:20*	9/13/2015 7:00*	7/4/2015 5:50
7/4/2015 9:40	9/14/2015 2:00	10/13/2015 8:15*
9/13/2015 7:00*	1/11/2016 14:40	9/14/2015 1:10*
1/11/2016 1:30*	1/11/2016 13:50	11/2/2015 9:56
7/4/2015 9:50	1/11/2016 12:00*	7/4/2015 6:30
9/16/2015 10:50*	1/11/2016 13:00	7/4/2015 4:30
9/14/2015 14:10	9/16/2015 10:50*	1/2/2016 9:20*
9/14/2015 13:50	9/17/2015 11:30	9/14/2015 2:00
7/4/2015 5:20	10/3/2015 14:40	9/14/2015 8:10*
9/14/2015 1:10*	1/2/2016 21:40	7/4/2015 4:20
1/12/2016 11:40	4/16/2015 23:10	1/11/2016 1:30*
1/12/2016 11:30*	10/4/2015 3:10	1/2/2016 21:40
9/14/2015 13:20	10/13/2015 8:15*	7/4/2015 4:40
7/4/2015 4:50	10/14/2015 23:35	9/16/2015 10:50*
9/14/2015 8:10*	10/14/2015 23:15	1/2/2016 1:30*
4/16/2015 23:10	1/2/2016 9:30*	1/11/2016 14:40
4/16/2015 16:00	4/16/2015 16:00	1/2/2016 9:10*
10/13/2015 8:15*	11/2/2015 9:56	1/12/2016 11:20*
7/4/2015 5:30	1/11/2016 1:30*	1/9/2016 6:50*
7/4/2015 9:10	1/11/2016 11:50	9/14/2015 13:05
.....	.....	.....
9/13/2015 19:10	10/13/2015 5:45	7/4/2015 0:00
7/4/2015 4:40	1/2/2016 21:00	7/4/2015 5:30
7/4/2015 6:20	1/9/2016 18:20	7/4/2015 6:20
7/4/2015 5:00	1/9/2016 18:40	7/4/2015 6:50
7/4/2015 13:50	9/14/2015 0:40	7/4/2015 7:00
.....	.....	.....
9/14/2015 8:00	9/14/2015 2:10	7/4/2015 7:50
1/9/2016 18:30	9/14/2015 8:20	10/13/2015 5:45
1/11/2016 13:00	9/14/2015 8:30	9/16/2015 11:00
1/11/2016 11:50	9/14/2015 8:40	10/13/2015 6:35
7/4/2015 9:30	10/14/2015 8:15	10/13/2015 8:25
.....	.....	.....
10/14/2015 7:25	1/9/2016 18:00	10/4/2015 4:30
10/14/2015 7:35	1/11/2016 11:40	10/4/2015 4:20
7/4/2015 10:10	10/13/2015 6:35	1/2/2016 21:50
7/4/2015 10:20	10/4/2015 23:10	1/11/2016 13:50
7/4/2015 10:30	9/13/2015 19:30	9/13/2015 21:40
.....	.....	.....
7/4/2015 10:50	1/9/2016 18:10	1/11/2016 12:10
1/11/2016 13:40	1/11/2016 13:40	1/9/2016 18:30
1/11/2016 13:50	9/13/2015 19:40	10/4/2015 3:10
10/4/2015 3:10	10/14/2015 7:55	1/11/2016 11:30
1/9/2016 18:40	1/11/2016 11:30	10/14/2015 7:25

Source: Ahmed et al. [4]. With permission.

R package `rpart` with the package's default parameter values. The resulting CART is in fact presented in Fig. 12.3, and the interpretation of the tree model and how it helps with fault diagnosis is discussed in Section 12.2.1.

GLOSSARY

ANN: Artificial neural network

CART: Classification and regression tree

CD: Connectivity-based distance, or chaining distance

COF: Connectivity-based outlier factor

CSV: Comma-separated values Excel file format

FABOD: Fast angle-based outlier detection

INFLO: Influenced outlierness

KDEOS: Kernel density estimation outlier score

kNN: k-th nearest neighbor distance-based anomaly detection

kNNW: k nearest neighborhood distances summation

LDF: Local density factor

LDOF: Local distance-based outlier factor

LOF: Local outlier factor

LoMST: Local minimum spanning tree

LoOP: Local outlier probabilities

LRD: Local reachability density

MARS: Multivariate adaptive regression splines

MD: Mahalanobis distance or statistical distance

MST: Minimum spanning tree

ODIN: Outlier detection using indegree number

PCA: Principal component analysis

RD: Reachability distance

RKHS: Reproducing kernel Hilbert space

SLOF: Simplified local outlier factor

SOD: Subspace outlying degree

SPC: Statistical process control

SQC: Statistical quality control

SVM: Support vector machine

EXERCISES

12.1 Speaking of the types of anomaly, one type of anomaly is called the contextual anomaly, meaning that an observation may be an anomaly when its covariates take certain values, but the same observation may not be an anomaly when its covariates are under a different condition. Please come up with some examples explaining the contextual anomaly.

12.2 Given Eq. 12.4, derive the expression of Eq. 12.5. Also prove that the eigenvector of $\mathbf{\Sigma}_{\boldsymbol{y}}$ remains $\mathbf{h}_i$ if the covariance matrix of $\boldsymbol{\varepsilon}$ is $\sigma_{\varepsilon}^2\mathbf{I}$.

12.3 Consider the p-norm in Section 12.3.1.

a. Recall that $\|\boldsymbol{x}\|_0$ is used to represent the number of nonzero elements in $\boldsymbol{x}$. Show that the 0-norm is not a valid norm. Which requirement for a valid norm is not satisfied by the 0-norm?

b. Prove that

$$\|\boldsymbol{x}\|_\infty \leq \|\boldsymbol{x}\|_2 \leq \|\boldsymbol{x}\|_1 \leq \sqrt{n}\|\boldsymbol{x}\|_2 \leq n\|\boldsymbol{x}\|_\infty,$$

where n is the dimension of $\boldsymbol{x}$.

12.4 One kernel function used in SVM or other machine learning methods is the d^{th}-degree polynomial kernel, defined as

$$K(\boldsymbol{x}, \boldsymbol{x}') = (1 + \langle \boldsymbol{x}, \boldsymbol{x}' \rangle)^d.$$

For the polynomial kernel, it is possible to write down the mapping functions, $\boldsymbol{\phi}(\boldsymbol{x})$, explicitly and in a closed form, so that the relationship,

$$K(\boldsymbol{x}, \boldsymbol{x}') = \langle \boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{x}') \rangle,$$

can be proven. Please do this for $d = 2$ by writing explicitly the expression of $K(\boldsymbol{x}, \boldsymbol{x}')$ and that of $\boldsymbol{\phi}(\boldsymbol{x})$, thereby showing and confirming the equality in the above equation.

12.5 To construct the plot of Fig. 12.6, consider a bivariate Gaussian distribution, $\mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$, where

$$\boldsymbol{\mu} = \begin{pmatrix} 5 \\ 5 \end{pmatrix}, \quad \text{and}$$
$$\mathbf{\Sigma} = \begin{pmatrix} 1 & 0.45 \\ 0.45 & 0.25 \end{pmatrix}.$$

The position of point A is $\boldsymbol{x}_A = (4, 7)^T$ and that of point B is $\boldsymbol{x}_B = (7, 6)^T$. The mean of the Gaussian distribution is represented by point C, whose position is $\boldsymbol{\mu} = (5, 5)^T$.

a. Please compute both the Euclidean and statistical distances of AC and BC. Do your results confirm the statement in Section 12.3.3?

b. Please compute the statistical distance between any point on the 99% probability contour to point C.

c. Can you drive the general formula for the statistical distance between any point on the $100(1-\alpha)\%$ probability contour and point C for a given $\alpha \in [0, 1]$?

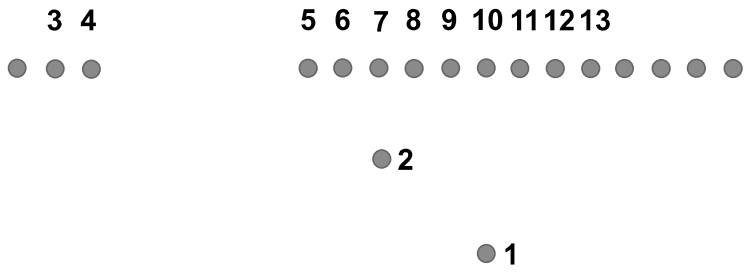
12.6 Can you please come up with an example illustrating a circumstance for which the kNN anomaly detection cannot correctly identify the anomaly but kNNW could?

12.7 Consider the anomaly examples presented in Fig. 12.1. Let us remove points, A_1 and A_2, and the cluster of C_3 for the moment. Then, imagine that the points in C_2 are clustered more tightly than those in C_1. Suppose that the points in both C_1 and C_2 are uniformly scattered. The nearest neighbor distance in C_1 is five (whatever unit it may be), whereas the nearest neighbor distance in C_2 is one. The distance between A_3 and its nearest neighbor in C_2 is three. Consider $k = 1$ and $k = 2$.

a. Use this example to show that both kNN and kNNW anomaly detection methods are ineffective to flag A_3 as an anomaly.

b. Show that LOF is capable of detecting A_3 as an anomaly.

12.8 Consider an example presented by Tang et al. [207]. Figure 3 in [207] is modified and then presented below, where the distance between points #1 and #2 is five, that between #2 and #7 is three, that between #4 and #5 is six, and the distance between any other two adjacent points on the line is one.



a. Let $k = 10$, and use this example to compute the chaining distance for point #1. Please show explicitly the sets of $\mathcal{G}$ and $\mathcal{E}$ associated with point #1.

b. Select a k and show that $\mathcal{G}$ could be different from $\mathfrak{N}$ for point #1.

12.9 Prove that in Eq. 12.17, the coefficients are summed to one when $D(\boldsymbol{x}_{(i-1)}, \boldsymbol{x}_{(i)})$ is the same for all i's.

12.10 In Section 12.4.4, we state that "[T]he danger of using a subspace approach is that if not chosen properly, the difference between a potential anomaly and normal points may disappear altogether in the subspace." One popular method to select a subspace is principal component analysis (PCA). PCA is to find the subspaces that account for the largest variance in data. Please come up with an example in a two-dimensional space, such that once the one-dimensional subspace of the largest variance is selected and data projected onto that space, the intrinsic structure existing in the original data disappears. In other words, the otherwise distinguishable two classes of data in the original two-dimensional data space are no longer separable in the wrongly selected one-dimensional subspace.

12.11 In Fig. 12.8, there are sixteen options of spanning trees that can connect all nodes without forming a cycle. Please list all sixteen choices and show that the one presented in the right panel is indeed the minimum spanning tree.

Bibliography

[1] T. Ackermann. *Wind Power in Power Systems*. John Wiley & Sons, New York, 2005.

[2] M. S. Adaramola and P.-Å. Krogstad. Experimental investigation of wake effects on wind turbine performance. *Renewable Energy*, 36(8):2078–2086, 2011.

[3] P. Agarwal and L. Manuel. Extreme loads for an offshore wind turbine using statistical extrapolation from limited field data. *Wind Energy*, 11:673–684, 2008.

[4] I. Ahmed, A. Dagnino, and Y. Ding. Unsupervised anomaly detection based on minimum spanning tree approximated distance measures and its application to hydropower turbines. *IEEE Transactions on Automation Science and Engineering*, 16(2):654–667, 2019.

[5] D. Aigner, C. A. K. Lovell, and P. Schmidt. Formulation and estimation of stochastic frontier production function models. *Journal of Econometrics*, 6(1):21–37, 1977.

[6] P. Ailliot and V. Monbet. Markov-switching autoregressive models for wind time series. *Environmental Modelling & Software*, 30:92–101, 2012.

[7] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

[8] T. W. Anderson. *The Statistical Analysis of Time Series*. John Wiley & Sons, New York, 1971.

[9] F. Angiulli and C. Pizzuti. Outlier mining in large high-dimensional datasets. *IEEE Transactions on Knowledge and Data Engineering*, 17(2):203–215, 2005.

[10] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM*, 45(6):891–923, 1998.

[11] R. D. Banker, A. Charnes, and W. W. Cooper. Some models for estimating technical and scale inefficiencies in data envelopment analysis. *Management Science*, 30(9):1078–1092, 1984.

[12] R. J. Barthelmie, K. S. Hansen, S. T. Frandsen, O. Rathmann, J. G. Schepers, W. Schlez, J. Phillips, K. Rados, A. Zervos, E. S. Politis, and P. K. Chaviaropoulos. Modelling and measuring flow and wind turbine wakes in large wind farms offshore. *Wind Energy*, 12(5):431–444, 2009.

[13] R. J. Barthelmie and L. E. Jensen. Evaluation of wind farm efficiency and wind turbine wakes at the Nysted offshore wind farm. *Wind Energy*, 13(6):573–586, 2010.

[14] R. J. Barthelmie, S. C. Pryor, S. T. Frandsen, K. S. Hansen, J. G. Schepers, K. Rados, W. Schlez, A. Neubert, L. E. Jensen, and S. Neckelmann. Quantifying the impact of wind turbine wakes on power output at offshore wind farms. *Journal of Atmospheric and Oceanic Technology*, 27(8):1302–1317, 2010.

[15] O. Belghazi and M. Cherkaoui. Pitch angle control for variable speed wind turbines using genetic algorithm controller. *Journal of Theoretical and Applied Information Technology*, 39:5–10, 2012.

[16] J. L. Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, 1975.

[17] R. J. Bessa, V. Miranda, A. Botterud, J. Wang, and E. M. Constantinescu. Time adaptive conditional kernel density estimation for wind power forecasting. *IEEE Transactions on Sustainable Energy*, 3(4):660–669, 2012.

[18] A. Betz. *Introduction to the Theory of Flow Machines*. Pergamon Press, 1966.

[19] F. D. Bianchi, H. De Battista, and R. J. Mantz. *Wind Turbine Control Systems*. Springer-Verlag, London, 2007.

[20] G. E. P. Box, G. M. Jenkins, and G. C. Reinsel. *Time Series Analysis: Forecasting and Control*. John Wiley & Sons, New York, 4th edition, 2008.

[21] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[22] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. LOF: Identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, volume 29, pages 93–104. ACM, 2000.

[23] B. G. Brown, R. W. Katz, and A. H. Murphy. Time series models to simulate and forecast wind speed and wind power. *Journal of Climate and Applied Meteorology*, 23:1184–1195, 1984.

[24] E. Byon. Wind turbine operations and maintenance: A tractable approximation of dynamic decision-making. *IIE Transactions*, 45:1188–1201, 2013.

[25] E. Byon and Y. Ding. Season-dependent condition-based maintenance for a wind turbine using a partially observed Markov decision process. *IEEE Transactions on Power Systems*, 25:1823–1834, 2010.

[26] E. Byon, L. Ntaimo, and Y. Ding. Optimal maintenance strategies for wind turbine systems under stochastic weather conditions. *IEEE Transactions on Reliability*, 59:393–404, 2010.

[27] E. Byon, L. Ntaimo, C. Singh, and Y. Ding. Wind energy facility reliability and maintenance. In P. M. Pardalos, S. Rebennack, M. V. F. Pereira, N. A. Iliadis, and V. Pappu, editors, *Handbook of Wind Power Systems: Optimization, Modeling, Simulation and Economic Aspects*, pages 639–672. Springer, Berlin Heidelberg, 2013.

[28] E. Byon, E. Pérez, Y. Ding, and L. Ntaimo. Simulation of wind farm operations and maintenance using DEVS. *Simulation–Transactions of the Society for Modeling and Simulation International*, 87:1093–1117, 2011.

[29] E. Byon, A. K. Shrivastava, and Y. Ding. A classification procedure for highly imbalanced class sizes. *IIE Transactions*, 42(4):288–303, 2010.

[30] G. O. Campos, A. Zimek, J. Sander, R. J. G. B. Campello, B. Micenková, E. Schubert, I. Assent, and M. E. Houle. On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery*, 30(4):891–927, 2016.

[31] J. A. Carta, P. Ramírez, and S. Velázquez. Influence of the level of fit a density probability function to wind-speed data on the WECS mean power output estimation. *Energy Conversion and Management*, 49:2647–2655, 2008.

[32] J. A. Carta, P. Ramírez, and S. Velázquez. A review of wind speed probability distributions used in wind energy analysis—case studies in the Canary Islands. *Renewable and Sustainable Energy Reviews*, 13:933–955, 2009.

[33] A. Carvalho, M. C. Gonzalez, P. Costa, and A. Martins. Issues on performance of wind systems derived from exploitation data. In *Proceedings of the 35th Annual Conference of IEEE Industrial Electronics*, pages 3599–3604, Porto, Portugal, 2009.

[34] J. E. Cavanaugh. Unifying the derivations of the Akaike and corrected Akaike information criteria. *Statistics and Probability Letters*, 31:201–208, 1997.

[35] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[36] H. A. Chipman, E. I. George, and R. E. McCulloch. BART: Bayesian additive regression trees. *The Annals of Applied Statistics*, 4:266–298, 2010.

[37] Y. Choe, E. Byon, and N. Chen. Importance sampling for reliability evaluation with stochastic simulation models. *Technometrics*, 57:351–361, 2015.

[38] S. G. Coles. *An Introduction to Statistical Modeling of Extreme Values*. Springer, New York, 2001.

[39] N. Conroy, J. P. Deane, and B. P. Ó. Gallachóir. Wind turbine availability: Should it be time or energy based? A case study in Ireland. *Renewable Energy*, 36(11):2967–2971, 2011.

[40] A. Crespo, J. Hernández, and S. Frandsen. Survey of modelling methods for wind turbine wakes and wind farms. *Wind Energy*, 2(1):1–24, 1999.

[41] N. A. C. Cressie. *Statistics for Spatial Data*. John Wiley & Sons, New York, 1991.

[42] N. A. C. Cressie, S. Burden, W. Davis, P. Krivitsky, P. Mokhtarian, T. Suesse, and A. Zammit-Mangion. Capturing multivariate spatial dependence: Model, estimate and then predict. *Statistical Science*, 30(2):170–175, 2015.

[43] N. A. C. Cressie and C. K. Wikle. *Statistics for Spatio-Temporal Data*. John Wiley & Sons, New York, 2011.

[44] P. Crochet. Adaptive Kalman filtering of 2-metre temperature and 10-metre wind-speed forecasts in Iceland. *Meteorological Applications*, 11(2):173–187, 2004.

[45] P. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134:19–67, 2005.

[46] D. G. T. Denison, C. C. Holmes, B. K. Mallick, and A. F. M. Smith. *Bayesian Methods for Nonlinear Classification and Regression*. John Wiley & Sons, New York, 2002.

[47] D. G. T. Denison, B. K. Mallick, and A. F. M Smith. Bayesian MARS. *Statistics and Computing*, 8:337–346, 1998.

[48] M. Derby. DOE's perspective. In *The 2011 National Renewable Energy Laboratory Workshop on Wind Turbine Condition Monitoring*, Broomfield, CO, 2011. September 19.

[49] Y. Ding, D. Ceglarek, and J. Shi. Fault diagnosis of multi-station manufacturing processes by using state space approach. *Transactions of ASME, Journal of Manufacturing Science and Engineering*, 124:313–322, 2002.

[50] Y. Ding, J. Tang, and J. Z. Huang. Data analytics methods for wind energy applications. In *Proceedings of ASME Turbo Expo 2015: Turbine Technical Conference and Exposition, GT2015-43286*, pages 1–9, Montreal, Canada, 2015. June 15–19.

[51] A. G. Drachmann. Heron's windmill. *Centaurus*, 7:145–151, 1961.

[52] S. D. Dubey. Normal and Weibull distributions. *Naval Research Logistics Quarterly*, 14:69–79, 1967.

[53] V. Dubourg, B. Sudret, and F. Deheeger. Metamodel-based importance sampling for structural reliability analysis. *Probabilistic Engineering Mechanics*, 33:47–57, 2013.

[54] J. Durbin. The fitting of time-series models. *Review of the International Statistical Institute*, 28(3):233–244, 1960.

[55] B. Efron and R. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall/CRC Press, New York, 1993.

[56] A. Emami and P. Noghreh. New approach on optimization in placement of wind turbines within wind farm by genetic algorithms. *Renewable Energy*, 35(7):1559–1564, 2010.

[57] Y. Ephraim and N. Merhav. Hidden Markov processes. *IEEE Transactions on Information Theory*, 48:1518–1569, 2002.

[58] B. Everitt. *The Cambridge Dictionary of Statistics*. Cambridge University Press, Cambridge, UK, 1998.

[59] A. A. Ezzat, M. Jun, and Y. Ding. Spatio-temporal asymmetry of local wind fields and its impact on short-term wind forecasting. *IEEE Transactions on Sustainable Energy*, 9(3):1437–1447, 2018.

[60] A. A. Ezzat, M. Jun, and Y. Ding. Spatio-temporal short-term wind forecast: A calibrated regime-switching method. *The Annals of Applied Statistics*, in press, 2019.

[61] J. Fan and T. H. Yim. A cross-validation method for estimating conditional densities. *Biometrika*, 91:819–834, 2004.

[62] F. Felker. The status and future of wind energy. Technical report, National Wind Technology Center, Boulder, CO, 2009. Available at `http://www.ncsl.org/documents/energy/Felker0609.pdf`.

[63] L. M. Fitzwater, C. A. Cornell, and P. S. Veers. Using environmental contours to predict extreme events on wind turbines. In *Proceedings of the 2003 ASME Wind Energy Symposium, WIND2003-865*, pages 244–285, 2003. Reno, Nevada.

[64] R. Fletcher. *Practical Methods of Optimization.* John Wiley & Sons, New York, 2th edition, 1987.

[65] J. Fogle, P. Agarwal, and L. Manuel. Towards an improved understanding of statistical extrapolation for wind turbine extreme loads. *Wind Energy*, 11:613–635, 2008.

[66] S. Fortunato. Community detection in graphs. *Physics Reports*, 486:75–174, 2010.

[67] K. Freudenreich and K. Argyriadis. Wind turbine load level based on extrapolation and simplified methods. *Wind Energy*, 11:589–600, 2008.

[68] J. Friedman. Multivariate adaptive regression splines. *Annals of Statistics*, 19(1):1–67, 1991.

[69] M. Fuentes. A high frequency kriging approach for non-stationary environmental processes. *Environmetrics*, 12(5):469–483, 2001.

[70] P. M. O. Gebraad, F. W. Teeuwisse, J. W. Wingerden, P. A. Fleming, S. D. Ruben, J. R. Marden, and L. Y. Pao. Wind plant power optimization through yaw control using a parametric model for wake effects—a CFD simulation study. *Wind Energy*, 19(1):95–114, 2016.

[71] G. Giebel, R. Brownsword, G. Kariniotakis, M. Denhard, and C. Draxl. The state-of-the-art in short-term prediction of wind power: A literature overview. Technical report, Risø National Laboratory, Roskilde, Denmark, 2011. Available at `http://www.anemos-plus.eu/images/pubs/deliverables/aplus.deliverable\_d1.2.stp\_sota\_v1.1.pdf`.

[72] T. Gneiting. Nonseparable, stationary covariance functions for space-time data. *Journal of the American Statistical Association*, 97(458):590–600, 2002.

[73] T. Gneiting. Making and evaluating point forecasts. *Journal of the American Statistical Association*, 106:746–762, 2011.

[74] T. Gneiting, M. Genton, and P. Guttorp. Geostatistical space-time models, stationarity, separability and full symmetry. In B. Finkenstadt, L. Held, and V. Isham, editors, *Statistical Methods for Spatio-Temporal Systems*, chapter 4. Chapman & Hall/CRC, 2007.

[75] T. Gneiting, K. Larson, K. Westrick, M. G. Genton, and E. Aldrich. Calibrated probabilistic forecasting at the Stateline wind energy center: The regime-switching space-time method. *Journal of the American Statistical Association*, 101:968–979, 2006.

[76] T. Gneiting and A. E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102:359–378, 2007.

[77] M. Goldstein and S. Uchida. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PLoS ONE*, 11(4):e0152173:1–31, 2016.

[78] P. J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.

[79] C. Gu. *Smoothing Spline ANOVA*. Springer-Verlag, New York, 2013.

[80] H. Guo, S. Watson, P. Tavner, and J. Xiang. Reliability analysis for wind turbines with incomplete failure data collected from after the date of initial installation. *Reliability Engineering and System Safety*, 94:1057–1063, 2009.

[81] S. T. Hackman. *Production Economics: Integrating the Microeconomic and Engineering Perspectives*. Springer-Verlag, Heidelberg, 2008.

[82] B. Hahn, M. Durstewitz, and K. Rohrig. Reliability of wind turbines—experiences of 15 years with 1,500 WTs. In J. Peinke, P. Schaumann, and S. Barth, editors, *Wind Energy: Proceedings of the Euromech Colloquium*, pages 329–332. Springer, 2007.

[83] P. Hall, J. Racine, and Q. Li. Cross-validation and the estimation of conditional probability. *Journal of the American Statistical Association*, 99:154–163, 2004.

[84] P. Hall and L. Simar. Estimating a changepoint, boundary, or frontier in the presence of observation error. *Journal of the American Statistical Association*, 97(458):523–534, 2002.

[85] K. S. Hansen, R. J. Barthelmie, L. E. Jensen, and A. Sommer. The impact of turbulence intensity and atmospheric stability on power deficits due to wind turbine wakes at Horns Rev wind farm. *Wind Energy*, 15(1):183–196, 2012.

[86] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, New York, 2nd edition, 2009.

[87] T. J. Hastie and R. J. Tibshirani. *Generalized Additive Models*. Chapman & Hall/CRC, 1990.

[88] M. He, L. Yang, J. Zhang, and V. Vittal. A spatio-temporal analysis approach for short-term forecast of wind farm generation. *IEEE Transactions on Power Systems*, 29(4):1611–1622, 2014.

[89] P. Heidelberger. Fast simulation of rare events in queueing and reliability models. *ACM Transactions on Modeling and Computer Simulation*, 5:43–85, 1995.

[90] E. J. Henley and H. Kumamoto. *Reliability Engineering and Risk Assessment*. Prentice-Hall, 1981.

[91] A. S. Hering and M. G. Genton. Powering up with space-time wind forecasting. *Journal of the American Statistical Association*, 105:92–104, 2010.

[92] C. Hildreth. Point estimates of ordinates of concave functions. *Journal of the American Statistical Association*, 49(267):598–619, 1954.

[93] D. Hinkley. On quick choice of power transformation. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 26(1):67–69, 1977.

[94] T. Hofmann, B. Schlkopf, and A. J. Smola. Kernel methods in machine learning. *Annals of Statistics*, 36(3):1171–1220, 2008.

[95] H. Hwangbo, Y. Ding, O. Eisele, G. Weinzierl, U. Lang, and G. Pechlivanoglou. Quantifying the effect of vortex generator installation on wind power production: An academia-industry case study. *Renewable Energy*, 113:1589–1597, 2017.

[96] H. Hwangbo, A. L. Johnson, and Y. Ding. A production economics analysis for quantifying the efficiency of wind turbines. *Wind Energy*, 20:1501–1513, 2017.

[97] H. Hwangbo, A. L. Johnson, and Y. Ding. Power curve estimation: Functional estimation imposing the regular ultra passum law. *Working Paper*, 2018. Available at SSRN: http://ssrn.com/abstract=2621033.

[98] H. Hwangbo, A. L. Johnson, and Y. Ding. Spline model for wake effect analysis: Characteristics of single wake and its impacts on wind turbine power generation. *IISE Transactions*, 50(2):112–125, 2018.

[99] B. J. Hyndman, D. M. Bashtannyk, and G. K. Grunwald. Estimating and visualizing conditional densities. *Journal of Computational and Graphical Statistics*, 5:315–336, 1996.

[100] International Electrotechnical Commission (IEC). *IEC TS 61400-1 Ed. 2: Wind Turbines – Part 1: Design Requirements*. IEC, Geneva, Switzerland, 1999.

[101] International Electrotechnical Commission (IEC). *IEC TS 61400-1 Ed. 3, Wind Turbines – Part 1: Design Requirements*. IEC, Geneva, Switzerland, 2005.

[102] International Electrotechnical Commission (IEC). *IEC TS 61400-12-1 Ed. 1, Wind Turbines – Part 12-1: Power Performance Measurements of Electricity Producing Wind Turbines*. IEC, Geneva, Switzerland, 2005.

[103] International Electrotechnical Commission (IEC). *IEC TS 61400-26-1 Ed. 1, Wind Turbines – Part 26-1: Time-based Availability for Wind Turbine Generating Systems*. IEC, Geneva, Switzerland, 2011.

[104] International Electrotechnical Commission (IEC). *IEC TS 61400-12-2 Ed. 1, Wind Turbines – Part 12-2: Power Performance of Electricity Producing Wind Turbines Based on Nacelle Anemometry*. IEC, Geneva, Switzerland, 2013.

[105] International Electrotechnical Commission (IEC). *IEC TS 61400-26-2 Ed. 1, Wind Turbines – Part 26-2: Production-based Availability for Wind Turbines*. IEC, Geneva, Switzerland, 2014.

[106] S. R. Jammalamadaka and A. SenGupta. *Topics in Circular Statistics*. World Scientific, 2001.

[107] A. H. Jazwinski. Adaptive filtering. *Automatica*, 5:475–485, 1969.

[108] N. O. Jensen. A note on wind generator interaction. Technical report Risø-M, No. 2411, Risø National Laboratory, Roskilde, Denmark, 1983. Available at `http://orbit.dtu.dk/files/55857682/ris_m_2411.pdf`.

[109] J. Jeon and J. W. Taylor. Using conditional kernel density estimation for wind power density forecasting. *Journal of the American Statistical Association*, 107:66–79, 2012.

[110] P. Jirutitijaroen and C. Singh. The effect of transformer maintenance parameters on reliability and cost: A probabilistic model. *Electric Power System Research*, 72:213–234, 2004.

[111] I. T. Jolliffe. *Principal Component Analysis*. Springer, New York, 2nd edition, 2002.

[112] B. J. Jonkman. *TurbSim User's Guide: Version 1.50*. National Renewable Energy Laboratory, Golden, CO, 2009.

[113] B. J. Jonkman and M. L. Buhl Jr. *FAST User's Guide*. National Renewable Energy Laboratory, Golden, CO, 2005.

[114] M. Jun and M. Stein. An approach to producing space-time covariance functions on spheres. *Technometrics*, 49(4):468–479, 2007.

[115] M. S. Kaiser, M. J. Daniels, K. Furakawa, and P. Dixon. Analysis of particulate matter air pollution using Markov random field models of spatial dependence. *Environmetrics*, 13:615–628, 2002.

[116] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of ASME, Journal of Basic Engineering*, 82(1):35–45, 1960.

[117] G. Kariniotakis. *Renewable Energy Forecasting: From Models to Applications*. Woodhead Publishing, 2017.

[118] R. E. Kass and L. Wasserman. A reference Bayesian test for nested hypotheses and its relationship to the Schwarz criterion. *Journal of the American Statistical Association*, 90:928–934, 1995.

[119] K. Kazor and A. S. Hering. The role of regimes in short-term wind speed forecasting at multiple wind farms. *Stat*, 4(1):271–290, 2015.

[120] N. D. Kelley and B. J. Jonkman. Overview of the TurbSim stochastic inflow turbulence simulator. NREL/TP-500-41137, Version 1.21, National Renewable Energy Laboratory, Golden, Colorado, 2003. Available at `https://nwtc.nrel.gov/system/files/TurbSimOverview.pdf`.

[121] M. G. Khalfallah and A. M. Koliub. Effect of dust on the performance of wind turbines. *Desalination*, 209(1):209–220, 2007.

[122] R. Killick and I. Eckley. Changepoint: An `R` package for changepoint analysis. *Journal of Statistical Software*, 58(3):1–19, 2014.

[123] J. Kjellin, F. Bülow, S. Eriksson, P. Deglaire, M. Leijon, and H. Bernhoff. Power coefficient measurement on a 12 kW straight bladed vertical axis wind turbine. *Renewable Energy*, 36(11):3050–3053, 2011.

[124] J. P. C. Kleijnen. *Design and Analysis of Simulation Experiments*. Springer-Verlag, New York, 2008.

[125] H.-P. Kriegel, P. Krger, E. Schubert, and A. Zimek. Outlier detection in axis-parallel subspaces of high dimensional data. In *Advances in Knowledge Discovery and Data Mining: Proceedings of the 13th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 831–838. Springer, Berlin Heidelberg, 2009.

[126] P.-Å. Krogstad and J. A. Lund. An experimental and numerical study of the performance of a model turbine. *Wind Energy*, 15(3):443–457, 2012.

[127] T. Kuosmanen. Representation theorem for convex nonparametric least squares. *The Econometrics Journal*, 11(2):308–325, 2008.

[128] A. Kusiak and Z. Song. Design of wind farm layout for maximum wind energy capture. *Renewable Energy*, 35(3):685–694, 2010.

[129] M. P. Laan, N. N. Sørensen, P.-E. Réthoré, J. Mann, M. C. Kelly, N. Troldborg, J. G. Schepers, and E. Machefaux. An improved k-ϵ model applied to a wind turbine wake in atmospheric turbulence. *Wind Energy*, 18(5):889–907, 2015.

[130] T. J. Larsen and A. M. Hansen. *How 2 HAWC2, the User's Manual.* Risø National Laboratory, Roskilde, Denmark, 2007.

[131] G. Lee, E. Byon, L. Ntaimo, and Y. Ding. Bayesian spline method for assessing extreme loads on wind turbines. *The Annals of Applied Statistics*, 7:2034–2061, 2013.

[132] G. Lee, Y. Ding, M. G. Genton, and L. Xie. Power curve estimation with multivariate environmental factors for inland and offshore wind farms. *Journal of the American Statistical Association*, 110(509):56–67, 2015.

[133] G. Lee, Y. Ding, L. Xie, and M. G. Genton. Kernel Plus method for quantifying wind turbine upgrades. *Wind Energy*, 18:1207–1219, 2015.

[134] G. Li and J. Shi. Application of Bayesian model averaging in modeling long-term wind speed distributions. *Renewable Energy*, 35:1192–1202, 2010.

[135] H. Link, W. LaCava, J. van Dam, B. McNiff, S. Sheng, R. Wallen, M. McDade, S. Lambert, S. Butterfield, and F. Oyague. Gearbox reliability collaborative project report: Findings from phase 1 and phase 2 testing. Technical report, National Renewable Energy Laboratory, Golden, CO, 2011. Available at `https://www.nrel.gov/docs/fy11osti/51885.pdf`.

[136] P. Louka, G. Galanis, N. Siebert, G. Kariniotakis, P. Katsafados, I. Pytharoulis, and G. Kallos. Improvements in wind speed forecasts for wind power prediction purposes using Kalman filtering. *Journal of Wind Engineering and Industrial Aerodynamics*, 96(12):2348–2362, 2008.

[137] W. Lovejoy. Computationally feasible bounds for partially observed Markov decision processes. *Operations Research*, 39:162–175, 1991.

[138] P. Lynch. *The Emergence of Numerical Weather Prediction: Richardson's Dream.* Cambridge University Press, Cambridge, UK, 2006.

[139] M. Maadooliat, J. Z. Huang, and J. Hu. Integrating data transformation in principal components analysis. *Journal of Computational and Graphical Statistics*, 24(1):84–103, 2015.

[140] P. C. Mahalanobis. On the generalized distance in statistics. *Proceedings of the National Institute of Sciences (Calcutta)*, 2:49–55, 1936.

[141] L. M. Maillart. Maintenance policies for systems with condition monitoring and obvious failures. *IIE Transactions*, 38:463–475, 2006.

[142] L. Manuel, P. S. Veers, and S. R. Winterstein. Parametric models for estimating wind turbine fatigue loads for design. *Transactions of ASME, Journal of Solar Energy Engineering*, 123:346–355, 2001.

[143] M. D. Marzio, A. Panzera, and C. C. Taylor. Smooth estimation of circular cumulative distribution functions and quantiles. *Journal of Nonparametric Statistics*, 24:935–949, 2012.

[144] M. D. Marzio, A. Panzera, and C. C. Taylor. Nonparametric regression for circular responses. *Scandinavian Journal of Statistics*, 40:238–255, 2013.

[145] M. D. Marzio, A. Panzera, and C. C. Taylor. Nonparametric regression for spherical data. *Journal of the American Statistical Association*, 109:748–763, 2014.

[146] P. McKay, R. Carriveau, and D. S.-K. Ting. Wake impacts on downstream wind turbine performance and yaw alignment. *Wind Energy*, 16(2):221–234, 2013.

[147] D. C. Montgomery. *Introduction to Statistical Quality Control.* John Wiley & Sons, New York, 6th edition, 2009.

[148] J. M. Morales, A. J. Conejo, H. Madsen, P. Pinson, and M. Zugno. *Integrating Renewables in Electricity Markets—Operational Problems.* Springer, New York, 2014.

[149] P. Moriarty. Database for validation of design load extrapolation techniques. *Wind Energy*, 11:559–576, 2008.

[150] P. Moriarty, W. E. Holley, and S. Butterfield. Effect of turbulence variation on extreme loads prediction for wind turbines. *Transactions of ASME, Journal of Solar Energy Engineering*, 124:387–395, 2002.

[151] H. Mueller-Vahl, G. Pechlivanoglou, C. N. Nayeri, and C. O. Paschereit. Vortex generators for wind turbine blades: A combined wind tunnel and wind turbine parametric study. In *Proceedings of ASME Turbo Expo 2012: Turbine Technical Conference and Exposition*, volume 6, pages 899–914, Copenhagen, Denmark, 2012. June 11–15.

[152] E. Nadaraya. On estimating regression. *Theory of Probability and Its Applications*, 9:141–142, 1964.

[153] A. Natarajan and W. E. Holley. Statistical extreme load extrapolation with quadratic distortions for wind turbines. *Transactions of ASME, Journal of Solar Energy Engineering*, 130:031017:1–7, 2008.

[154] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

[155] B. Niu, H. Hwangbo, L. Zeng, and Y. Ding. Evaluation of alternative efficiency metrics for offshore wind turbines and farms. *Renewable Energy*, 128:81–90, 2018.

[156] O. B. Olesen and J. Ruggiero. Maintaining the regular ultra passum law in data envelopment analysis. *European Journal of Operational Research*, 235(3):798–809, 2014.

[157] S. Øye. The effect of vortex generators on the performance of the ELKRAFT 1000 kW turbine. In *Aerodynamics of Wind Turbines: 9th IEA Symposium*, pages 9–14, Stockholm, Sweden, 1995. December 11–12.

[158] C. Pacot, D. Hasting, and N. Baker. Wind farm operation and maintenance management. In *Proceedings of the PowerGen Conference Asia*, pages 25–27, Ho Chi Minh City, Vietnam, 2003.

[159] C. Park, J. Z. Huang, and Y. Ding. A computable plug-in estimator of minimum volume sets for novelty detection. *Operations Research*, 58(5):1469–1480, 2010.

[160] J. M. Peeringa. Extrapolation of extreme responses of a multi-megawatt wind turbine. ECN-C-03-131, Energy Research Centre of the Netherlands, Petten, Netherlands, 2003. Available at `http://www.ecn.nl/docs/library/report/2003/c03131.pdf`.

[161] E. Pérez, Y. Ding, and L. Ntaimo. Multi-component wind turbine modeling and simulation for wind farm operations and maintenance. *Simulation–Transactions of the Society for Modeling and Simulation International*, 91:360–382, 2015.

[162] S. Pieralli, M. Ritter, and M. Odening. Efficiency of wind power production and its determinants. *Energy*, 90:429–438, 2015.

[163] P. Pinson. Wind energy: Forecasting challenges for its operational management. *Statistical Science*, 28:564–585, 2013.

[164] P. Pinson, L. Christensen, H. Madsen, P. E. Sørensen, M. H. Donovan, and L. E. Jensen. Regime-switching modelling of the fluctuations of offshore wind generation. *Journal of Wind Engineering and Industrial Aerodynamics*, 96(12):2327–2347, 2008.

[165] P. Pinson, H. A. Nielsen, H. Madsen, and T. S. Nielsen. Local linear regression with adaptive orthogonal fitting for wind power application. *Statistics and Computing*, 18:59–71, 2008.

[166] A. Pourhabib, J. Z. Huang, and Y. Ding. Short-term wind speed forecast using measurements from multiple turbines in a wind farm. *Technometrics*, 58(1):138–147, 2016.

[167] A. Pourhabib, B. K. Mallick, and Y. Ding. Absent data generating classifier for imbalanced class sizes. *Journal of Machine Learning Research*, 16:2695–2724, 2015.

[168] R. C. Prim. Shortest connection networks and some generalizations. *Bell Labs Technical Journal*, 36(6):1389–1401, 1957.

[169] J. M. Prospathopoulos, E. S. Politis, K. G. Rados, and P. K. Chaviaropoulos. Evaluation of the effects of turbulence model enhancements on wind turbine wake predictions. *Wind Energy*, 14(2):285–300, 2011.

[170] M. Puterman. *Markov Decision Processes*. John Wiley & Sons, New York, 1994.

[171] A. E. Raftery. Bayesian model selection in social research. *Sociological Methodology*, 25:111–163, 1995.

[172] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large datasets. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, volume 29, pages 427–438. ACM, 2000.

[173] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[174] P. Regan and L. Manuel. Statistical extrapolation methods for estimating wind turbine extreme loads. *Transactions of ASME, Journal of Solar Energy Engineering*, 130:031011:1–15, 2008.

[175] S. Rehman and N. M. Al-Abbadi. Wind shear coefficients and their effect on energy production. *Energy Conversion and Management*, 46:2578–2591, 2005.

[176] G. Reikard. Using temperature and state transitions to forecast wind speed. *Wind Energy*, 11(5):431–443, 2008.

[177] J. Ribrant. *Reliability Performance and Maintenance—A Survey of Failures in Wind Power Systems*. Master's Thesis, School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, Stockholm, Sweden, 2006.

[178] M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In *Proceedings of the 1993 IEEE International Conference on Neural Networks*, San Francisco, CA, 1993. March 28–April 1.

[179] R. A. Rigby and D. M. Stasinopoulos. Generalized additive models for location, scale and shape. *Applied Statistics, Series C*, 54:507–554, 2005.

[180] Risø-DTU. Wind Turbine Load Data: `http://www.winddata.com`.

[181] D. Robb. Improved wind station profits through excellence in O&M. *Wind Stats Report*, 24:5–6, 2011.

[182] D. Robb. Wind technology: What is working and what is not? *Wind Stats Report*, 24:4–5, 2011.

[183] K. O. Ronold and G. C. Larsen. Reliability-based design of wind-turbine rotor blades against failure in ultimate loading. *Engineering Structures*, 22:565–574, 2000.

[184] M. Rosenblatt. Conditional probability density and regression estimates. In P. Krishnaiah, editor, *Multivariate Analysis II*, pages 25–31. Academic Process, New York, 1969.

[185] D. B. Rubin. Matching to remove bias in observational studies. *Biometrics*, 29(1):159–183, 1973.

[186] D. B. Rubin. Using propensity scores to help design observational studies: Application to the tobacco litigation. *Health Services and Outcomes Research Methodology*, 2:169–188, 2001.

[187] D. Ruppert, S. J. Sheather, and M. P. Wand. An effective bandwidth selector for local least squares regression. *Journal of the American Statistical Association*, 90:1257–1270, 1995.

[188] Y. Saatçi, R. Turner, and C. E. Rasmussen. Gaussian process change point models. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 927–934, Haifa, Israel, 2010. June 21–24.

[189] J. Sacks, S. B. Schiller, and W. J. Welch. Designs for computer experiments. *Technometrics*, 31:41–47, 1989.

[190] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn. Design and analysis of computer experiments. *Statistical Science*, 4:409–423, 1989.

[191] I. Sanchez. Short-term prediction of wind energy production. *International Journal of Forecasting*, 22:43–56, 2006.

[192] B. Sanderse, S. P. Pijl, and B. Koren. Review of computational fluid dynamics for wind turbine wake aerodynamics. *Wind Energy*, 14(7):799–819, 2011.

[193] T. J. Santner, B. J. Williams, and W. I. Notz. *The Design and Analysis of Computer Experiments*. Springer-Verlag, New York, 2003.

[194] M. Sathyajith. *Wind Energy: Fundamentals, Resource Analysis and Economics*. Springer, Berlin Heidelberg, 2006.

[195] M. Schlater. Some covariance models based on normal scale mixtures. *Bernoulli*, 16(3):780–797, 2010.

[196] B. Schlkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

[197] G. E. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

[198] Y. E. Shin, Y. Ding, and J. Z. Huang. Covariate matching methods for testing and quantifying wind turbine upgrades. *The Annals of Applied Statistics*, 12(2):1271–1292, 2018.

[199] G. L. Smith. Sequential estimation of observation error variances in a trajectory estimation problem. *AIAA Journal*, 5(11):1964–1970, 1967.

[200] R. L. Smith. Extreme value theory. In W. Ledermann, E. Lloyd, S. Vajda, and C. Alexander, editors, *Handbook of Applicable Mathematics*, volume 7, pages 437–472. John Wiley & Sons, 1990.

[201] Z. Song, Y. Jiang, and Z. Zhang. Short-term wind speed forecasting with Markov-switching model. *Applied Energy*, 130:103–112, 2014.

[202] J. D. Sørensen and S. R. K. Nielsen. Extreme wind turbine response during operation. *Journal of Physics: Conference Series*, 75:012074:1–7, 2007.

[203] I. Staffell and R. Green. How does wind farm performance decline with age? *Renewable Energy*, 66:775–786, 2014.

[204] M. Stein. Space-time covariance functions. *Journal of the American Statistical Association*, 100(469):310–321, 2005.

[205] B. Stephen, S. J. Galloway, D. McMillan, D. C. Hill, and D. G. Infield. A copula model of wind turbine performance. *IEEE Transactions on Power Systems*, 26:965–966, 2011.

[206] E. A. Stuart. Matching methods for causal inference: A review and a look forward. *Statistical Science*, 25(1):1–21, 2010.

[207] J. Tang, Z. Chen, A. W. Fu, and D. W. Cheung. Enhancing effectiveness of outlier detections for low density patterns. In *Advances in Knowledge Discovery and Data Mining: Proceedings of the Sixth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 535–548. Springer, Berlin Heidelberg, 2002.

[208] J. Tatsu, P. Pinson, P. Trombe, and H. Madsen. Probabilistic forecasts of wind power generation accounting for geographically dispersed information. *IEEE Transactions on Smart Grid*, 5(1):480–489, 2014.

[209] P. Tavner, S. Faulstich, B. Hahn, and G. J. W. van Bussel. Reliability and availability of wind turbine electrical and electronic components. *EPE Journal*, 20(4):45–50, 2010.

[210] P. J. Tavner, C. Edwards, A. Brinkman, and F. Spinato. Influence of wind speed on wind turbine reliability. *Wind Engineering*, 30:55–72, 2006.

[211] P. J. Tavner, J. Xiang, and F. Spinato. Reliability analysis for wind turbines. *Wind Energy*, 10:1–8, 2007.

[212] C. C. Taylor. Automatic bandwidth selection for circular density estimation. *Computational Statistics and Data Analysis*, 52:3493–3500, 2008.

[213] J. B. Tenenbaum, V. De Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

[214] J. L. Torres, A. García, M. De Blas, and A. De Francisco. Forecast of hourly average wind speed with ARMA models in Navarre Spain. *Solar Energy*, 79(1):65–77, 2005.

[215] N. Troldborg, G. C. Larsen, H. A. Madsen, K. S. Hansen, J. N. Sørensen, and R. Mikkelsen. Numerical simulations of wake interaction between two wind turbines at various inflow conditions. *Wind Energy*, 14(7):859–876, 2011.

[216] O. Uluyol, G. Parthasarathy, W. Foslien, and K. Kim. Power curve analytic for wind turbine performance monitoring and prognostics. In *Annual Conference of the Prognostics and Health Management Society*, volume 2, page 049, Montreal, Canada, 2011. August 19.

[217] U.S. Department of Energy. Wind Vision: A New Era for Wind Power in the United States: `http://www.energy.gov/sites/prod/files/WindVision_Report_final.pdf`.

[218] U.S. Department of Energy. 20% Wind energy by 2030—Increasing wind energy's contribution to U.S. electricity supply. DOE/GO-102008-2567, Office of Energy Efficiency and Renewable Energy, Washington, D.C., 2008. Available at `https://www.nrel.gov/docs/fy08osti/41869.pdf`.

[219] U.S. Energy Information Agency. Annual Energy Review 2011: `https://www.eia.gov/totalenergy/data/annual/pdf/aer.pdf`.

[220] U.S. Energy Information Agency. Electric Power Annual 2016, Chapter 1 National Summary Data: `https://www.eia.gov/electricity/annual/pdf/epa.pdf`.

[221] V. N. Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag, New York, 1995.

[222] H. R. Varian. The nonparametric approach to demand analysis. *Econometrica*, 50:945–973, 1982.

[223] P. S. Veers and S. Butterfield. Extreme load estimation for wind turbines: Issues and opportunities for improved practice. In *Proceedings of the 2001 ASME Wind Energy Symposium, AIAA-2001-0044*, Reno, Nevada, 2001.

[224] C. M. Velte, M. O. L. Hansen, K. E. Meyer, and P. Fuglsang. Evaluation of the performance of vortex generators on the DU 91-W2-250 profile using stereoscopic PIV. In *Proceedings of the 12th World Multi-Conference on Systemics, Cybernetics and Informatics (WMSCI 2008)*, volume 2, pages 263–267, Orlando, Florida, 2008. June 29-July 2.

[225] Y. Wan, M. Milligan, and B. Parsons. Output power correlation between adjacent wind power plants. *Transactions of the ASME, Journal of Solar Energy Engineering*, 125:551–555, 2003.

[226] G. Watson. Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics, Series A*, 26:359–372, 1964.

[227] J. Wen, Y. Zheng, and D. Feng. A review on reliability assessment for wind power. *Renewable and Sustainable Energy Reviews*, 13:2485–2494, 2009.

[228] S. N. Wood. Thin plate regression splines. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(1):95–114, 2003.

[229] Y. Xia, K. H. Ahmed, and B. W. Williams. Wind turbine power coefficient analysis of a new maximum power point tracking technique. *IEEE Transactions on Industrial Electronics*, 60(3):1122–1132, 2013.

[230] N. Yampikulsakul, E. Byon, S. Huang, and S. Sheng. Condition monitoring of wind turbine system with non-parametric regression-based analysis. *IEEE Transactions on Energy Conversion*, 29(2):288–299, 2014.

[231] J. Yan, K. Li, E. Bai, J. Deng, and A. Foley. Hybrid probabilistic wind power forecasting using temporally local Gaussian process. *IEEE Transactions on Sustainable Energy*, 7(1):87–95, 2016.

[232] M. You, E. Byon, J. Jin, and G. Lee. When wind travels through turbines: A new statistical approach for characterizing heterogeneous wake

effects in multi-turbine wind farms. *IISE Transactions*, 49(1):84–95, 2017.

[233] B. Zeigler, T. Kim, and H. Praehofer. *Theory of Modeling and Simulation*. Academic Press, Orlando, FL, 2000.

[234] F. Zwiers and H. Von Storch. Regime-dependent autoregressive time series modeling of the Southern Oscillation. *Journal of Climate*, 3(12):1347–1363, 1990.

Index