# Chapter 11
# Probabilistic Forecast Methods


The previous two chapters were concerned with point forecasts which only produce a single estimate for each time step in the forecast horizon, i.e. one value $L_t$ for each of the time periods $t = N + 1, N + 2, \ldots, N + k$ (assuming a forecast horizon of length $k$ steps ahead starting at forecast origin $N$). Point estimates are limited in their description of the future demand, especially when the underlying data has a large degree of uncertainty. A more detailed picture of the possible values of the demand can be produced by estimating the *distribution* of the demand for each period in the forecast horizon. Forecasts which estimate the spread of the distribution are often called **probabilistic forecasts**. That is the subject of this chapter.

## 11.1 The Different Forms of Probabilistic Forecasts

As introduced in Sect. 5.2 and Fig. 5.4, there are three core forms of probabilistic forecasts which will be explored in this book: quantile forecasts, density forecasts and ensemble forecasts (not to be confused with ensemble machine learning models such as random forest in Sect. 10.3.2). These can be grouped into two core categories: **univariate** (quantile and density) and **multivariate** (the ensemble forecasts). To understand these types, consider the scenario of trying to estimate the distribution of the data for the time steps $t = N + 1, N + 2, \ldots, N + k$.

For a univariate forecast, the aim is to estimate the distribution of the demand at each time step. In other words, estimate a total of $k$ univariate distributions. The distributions at different time steps are independent of each other, meaning that the spread of the variable at one time step is not influenced by information about the demand at other time steps. Another way to say this is there are no *inter-dependencies* modelled. The univariate Gaussian, as introduced in Sect. 3.1 is the most famous example of a univariate distribution function. There are two main forms for estimates of univariate distributions, either a full continuous density function, or discrete values (quantiles)



S. Haben et al., *Core Concepts and Methods in Load Forecasting*,
https://doi.org/10.1007/978-3-031-27852-5_11

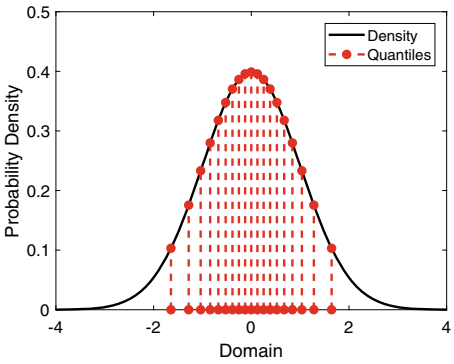


**Fig. 11.1** Example of both Gaussian distribution as described by its density function (solid line) and 20-quantiles (red dotted lines)

defining uniformly spaced levels of equal probability.[1] These two forms are illustrated in Fig. 11.1 for a standard Gaussian distribution. The density estimate is often preferable, as it describes the entire distribution, but requires either knowing/assuming that the distribution is from a particular parametric family (e.g. Gaussian in this case), or requires training relatively expensive methods, such as Kernel Density Estimates which will be described in Sect. 11.5. The quantile estimates (described in more detail in Sect. 3.2) in Fig. 11.1, show the 5, 10, . . . , 95% quantiles for the Gaussian density, and are clearly less descriptive of the distribution, however they are a lot less expensive to compute and don't rely on assuming a specific distribution of the data. Since a univariate distribution has to be estimated for each of the $k$ time steps in the forecast horizon this reduction in computational cost can be particularly advantageous.

For multivariate forecasts the task is instead to estimate a single multivariate distribution for all $k$ demand variables in the forecast horizon (see Sect. 3.3 for more on multivariate distributions). The advantage of multivariate distributions is that they take into account the *inter-dependencies* over the entire forecast time horizon. To illustrate this, consider the example of household demand. This is mainly determined by the occupants behaviour. If a person gets into work late then they will likely get back from work later, hence their demand shift in the morning will correspond to a shift in the evening. In other words, there is an **interdependency** between the demand in the morning and the demand in the evening due to the link between these two activities. A multivariate forecast can therefore be sampled to produce demand profile scenarios which include these correlations. Thus more complicated and realistic interdependent behaviours can be simulated and utilised to optimise applications such as storage control (Sect. 15.1).

Similar to the univariate case, the full multivariate density can be estimated but it is typically more complicated and difficult to model accurately. The methods are often more computationally expensive, there is fewer packages/resources for fitting them, and there are very few standard parametric multivariate distribution functions which can be used to fit to the data. Instead finite samples from the distribution are

[1] Actually the quantiles do not have to be uniformly spaced but it can often be simpler and more useful to do this.

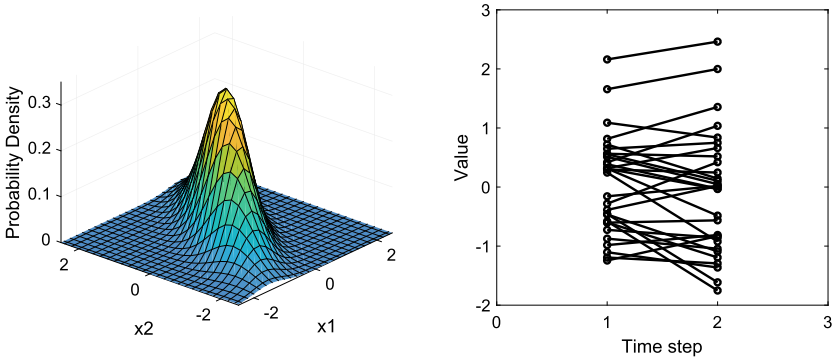


**Fig. 11.2** A bivariate Gaussian (left) and 30 ensembles from the distribution (right)

often estimated instead. These *ensembles* are realisations or 'representatives' from the distribution. To illustrate this consider a basic case with $k = 2$ where the estimates at two consecutive time steps $t = 1, 2$ are jointly described by a bivariate Gaussian distribution, shown in the left of Fig. 11.2. Drawing 30 random samples from this distribution gives the time-dependent correlated bivariate ensembles on the right. Notice in the language of distributions introduced in Sect. 3.3 that the multivariate probabilistic forecast is a joint distribution and the univariate probabilistic forecasts are marginal distributions of the full joint distribution.

## 11.2 Estimating Future Distributions

As discussed in the previous section, the aim of a probabilistic forecast is to estimate the future distribution of the demand whether at a single time step (univariate) or multiple (multivariate). To estimate the uncertainty requires accurately modelling the variation. There are a few standard practical approaches, which will be outlined in the section, and are the basis for many of the techniques in the following sections.

The first approach tries to model the distribution directly by training on the observations. As with most point forecasts these models use the historical data to capture the variation and will typically make assumption about how the past distribution will relate to the future demand. The parameteric models (Sect. 11.3), kernel density estimation (Sect. 11.5 ) and the quantile regression (Sect. 11.4 ) all model the distribution in this way.

The aim is to train the parameters or hyperparameters of a distribution model directly (e.g. the Gaussian model) or use a model which will estimate the distribution (e.g. quantiles). The advantage of these approaches is that as long as the right model is used, and they are trained on sufficient data from the target distribution, then they can accurately capture the uncertainty. For example, if we are modeling demand for 2pm and we know that the historic 2pm data all come from the same distribution then this

data can be used to estimate the true distribution. Unfortunately, it is often not known for certain which data comes from the same distribution and so certain assumptions will need to be made based on the analysis of the data. Another drawback of this approach is that its accuracy is correlated with the amount of available data. Small amounts of data will mean a potentially inaccurate estimate.

The second type of model doesn't model the variation directly but instead inserts variation into the model either through adjusting the input variables and/or the model parameters. For example, assume there is a demand model which is dependent on temperature alone. Then the variation in the demand can be modelled by inserting different values of the temperature into the model. Usually these are formed by tweaks on an individual estimate of the temperature and simulates the sensitivity of the demand to the temperature.

This approach is used in numerical weather prediction to produce forecast ensembles/scenarios. Small deviations are applied to the most likely state of the atmosphere and the numerical weather prediction models are reapplied to the adjusted states to produce a range of weather scenarios. Analysis of closeness of the final ensembles can indicate confidence in the future weather states, and widely ranging ensembles may mean there the future weather is highly uncertain.

Alternatively small adjustments can be applied to the model parameters. This accounts for mis-specifications in the model and can generate other likely future states. Multiple adjustments can therefore produce a range of outputs allowing for an estimate of the future distribution. The difficultly with both of these adjustment approaches is that the correct deviations have to be applied to the inputs/parameters in order to produce an accurate distribution estimate. This can be aided in the input case by randomly sampling from the historical observations, or from estimating a distribution from which you can sample.

Another drawback of this model is that the demand variation is not being simulated directly but instead is estimating the sensitivity of the model to the inputs or parameters. Consider the temperature example above. The demand may change with the temperature but in fact it is the variation in the demand for a fixed temperature which is of primary interest (assuming the temperature can be accurately forecast). The key is to add adjustments to the temperature so that it captures this variation. Once again cross-validation is one approach which can be used to determine an appropriate adjustment to the inputs/parameters.

The following sections will mainly focus on the first approach for producing probabilistic forecasts and train the models directly on the historical observations.

### *11.2.1 Notation*

In the following subsections a few probabilistic forecast methodologies are introduced for at least one of each of the three types introduced in Sect. 11.1: quantile, density and ensemble forecast. For the next sections it is worth considering the following notation and conditions.

1. As before consider the demand is represented by the time series $L_1, L_2, \ldots$, where $L_t$ is the demand at time step $t$.
2. Without loss of generality suppose the aim is to forecast the demand $k$-steps ahead for the time stamps $t = N+1, N+2, \ldots, N+k$.
3. For univariate probabilistic forecasts: denote the true distributions as CDFs, $F_1(L_{N+1}|\mathbf{Z}), F_2(L_{N+2}|\mathbf{Z}), \ldots, F_k(L_{N+k}|\mathbf{Z})$, one function for each time step in the forecast horizon, i.e. $F_t$ is the univariate distribution of the demand at time step $t$. Each forecast is conditional on prior information $\mathbf{Z}$ which represents the set of all required dependent variables such as weather, historical demand etc. which determine the future demand. The corresponding CDF forecasts, for each time step $t \in \{1, 2, \ldots, k\}$, are denoted $\hat{F}_t(L_{N+t}|\mathbf{Z})$. For simplicity the $\mathbf{Z}$ may not be included in the notation.
4. For the multivariate probabilistic forecasts the true distribution can be represented by a single CDF, $F_{t=1,\ldots,k}(\mathbf{L}|\mathbf{Z})$ describing the distribution of the multivariate random variable $\mathbf{L} = (L_{N+1}, L_{N+2}, \ldots, L_{N+k})^T$. The prior information $\mathbf{Z}$ contains all dependent variables and the historical loads up to time step $N$. Often the $\mathbf{Z}$ will not be included for clarity.
5. The $m$th ensemble of an ensemble forecast will often be denote as $\hat{\mathbf{L}}^{(m)} = (\hat{L}^{(m)}_{N+1}, \hat{L}^{(m)}_{N+2}, \ldots, \hat{L}^{(m)}_{N+k})^T$.

## 11.3 Parametric Models

Parametric distribution models are desirable as they can give a full description of the spread of the data usually using only a few parameters. This section begins by discussing parametric models via a simple example of a univariate distribution (Sect. 11.3.1). Individual univariate parametric models are usually too inflexible to model the distributions accurately, but families of simple univariate distributions can be "mixed" to estimate much more general shapes and will be introduced in Sect. 11.3.2.

### *11.3.1 Simple Univariate Distributions*

Some simple univariate distributions have already been introduced in Sect. 3.1. The most common being the Gaussian (or Normal) distribution, but the lognormal, and the gamma distribution were also presented. A range of distributions can be modelled using these functions in addition to other similar ones. The advantage of such models is that they only require training a small number of parameters to fully estimate the distribution. However, the restriction to a specific functional form means simple parametric models cannot estimate more complex distributions. For example, the distribution functions mentioned above are all **unimodal** which means they describe distributions with a single modal (maximum) value. It would be impossible to model

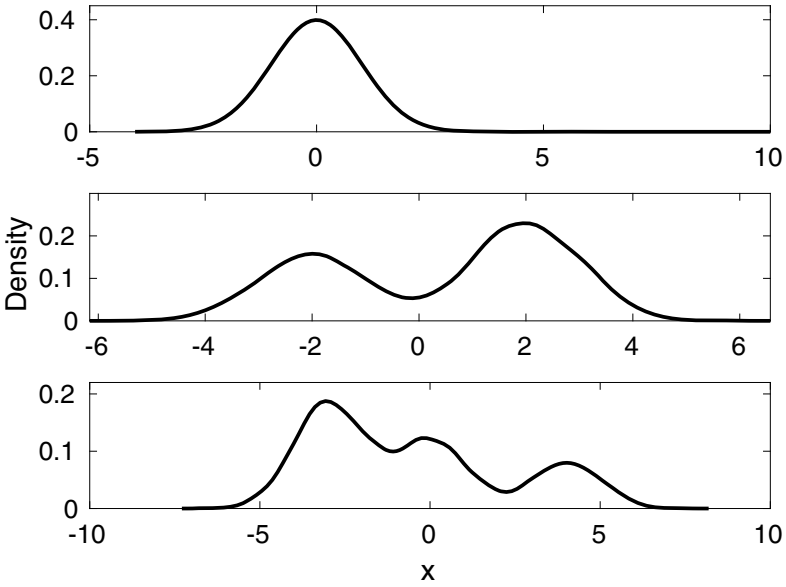


**Fig. 11.3** Examples of unimodal (top), bimodal (middle) and trimodal (bottom) univariate distributions

multimodal distributions (distributions with multiple distinct maximum values). An example of a unimodal, bimodal and trimodal distribution are shown in Fig. 11.3.

Although univariate models are unlikely to produce the most accurate univariate probabilistic forecasts they can be useful as benchmark models to compare to more sophisticated approaches described later in this chapter. Further since they are described by relatively few parameters they may be easier to train than non-parametric models. Training parametric models requires estimating each individual parameter which describes the chosen distribution family. For example, a Gaussian will require estimates for the mean and standard deviation, whereas the gamma distribution requires estimating the shape and scale parameters. In the case of the Gaussian distribution the mean and standard deviation can be found by maximum likelihood estimation (Sect. 8.2.1) and these values turn out to simply be the sample mean and sample standard deviation (Sect. 3.5) respectively. To ensure the best possible estimate is produced requires carefully selecting the most appropriate input data to train the parameters (in contrast to data driven machine learning techniques which will learn from all the data). The data can be identified by the analysis methods outlined in Chap. 6. For example, suppose some hourly data is discovered to have strong daily periodicity then it may be appropriate to train 24 different models, each one using only the data from a specific hour of the day.

Parametric models also exist for multivariate models. In particular there is a multivariate version of the Gaussian distribution. As mentioned in Sect. 11.1 these parametric models can be used to produce ensemble probabilistic forecasts over the

$h$-steps ahead (by estimating a $h$-dimensional parametric multivariate distribution). Unfortunately, there are much fewer well defined multivariate distributions which can accurately capture a wide variety of probabilistic forecast behaviours. This makes them less suitable compared to more versatile methods which will be introduced in Sects. 11.6 and 11.7 which can also capture interdependencies across time steps in the forecast horizon.

### *11.3.2 Mixture Models*

More versatile univariate distribution can be modelled by combining *mixtures* of the simple parametric distributions discussed in Sect. 11.3.1. The general form of a PDF for a **finite mixture model** of a random variable $\mathbf{x} \in \mathbb{R}^p$ is

$$f(\mathbf{x}) = \sum_{k=1}^{K} \pi_k g_k(\mathbf{x}, \theta_k), \tag{11.131}$$

where $g_k(\mathbf{x})$ are PDF's usually from a single family (e.g. Gaussian's) with their own corresponding parameter's $\theta_k$ (e.g. mean and standard deviation for a Gaussian). The $\pi_k$ are weights which satisfy $\sum_{k=1}^{K} \pi_k = 1$, and are often called **mixing probabilities**. Mixture models are often used for clustering, and in this case each PDF defines a distribution of points from one of the clusters, and the weights signify what proportion of the observations are in each cluster.

The most common mixture models for continuous variables use Gaussian components, i.e.

$$g_k(\mathbf{x}) = \frac{1}{(2\pi)^{N/2} det(\mathbf{\Sigma}_k)^{1/2}} exp\left((\mathbf{x} - \boldsymbol{\mu}_k)^T \mathbf{\Sigma}_k^{-1} (\mathbf{x} - \boldsymbol{\mu}_k)\right), \tag{11.132}$$

with covariance $\Sigma_k \in \mathbb{R}^{p \times p}$, and mean vector $\boldsymbol{\mu}_k \in \mathbb{R}^p$. This is called a Gaussian mixture model (GMM). A simple example of a GMM ($p = 1$) with three clusters is shown in Fig. 11.4 with mixture probabilities of 0.5, 0.25 and 0.25, means of 1, 3, 6 and all the same standard deviation of 1. It is easy to see that more complicated distributions can be estimated by adding more groups/clusters.

Although GMMs have a lot more parameters to fit to the observations then a single Gaussian model they can be solved relatively efficiently via an iterative process called the **expectation-maximisation algorithm** (EM) which finds an optimal estimate[2] for the maximum likelihood function (See Sect. 8.2.1).

Consider observations, $\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_N} \in \mathbb{R}^p$. Without going into the details of the EM-algorithm, the process iterates between an expectation step (E-step), which

[2] Note this is more than likely a locally optimal rather than a globally optimal estimate.

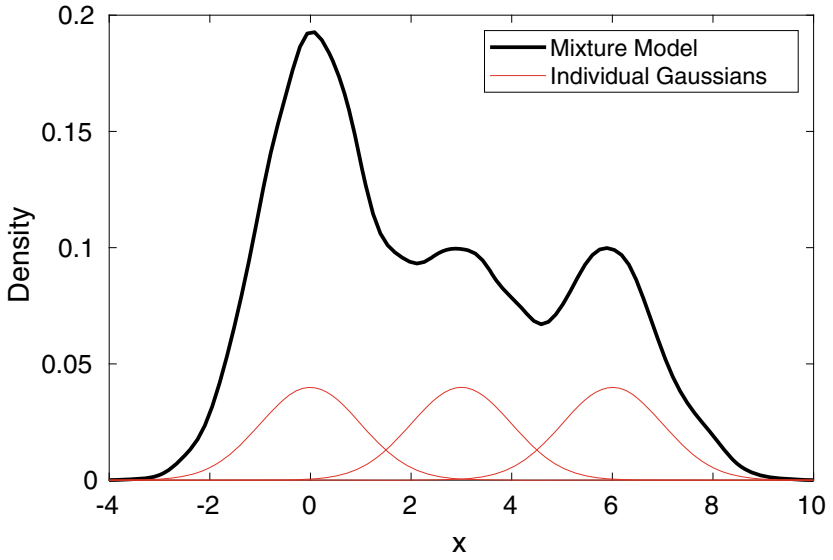


**Fig. 11.4** Example of a three component Gaussian mixture model. Also shown are scaled version of the individual Gaussian components (in red) to show their positions and how they contribute to the overall distribution of the GMM

calculates the expectation of the log-likelihood function with current estimates of the parameters, and the maximisation (M-step) which updates the parameters which maximises the current expected log-likelihood function. For a GMM this translates to the following steps (calculated for each iteration):

1. Calculate the posterior probability $\tau_{ik}$ that each observations $\mathbf{x}_i$ belongs to each group $k = 1, \ldots, K$,
$$\tau_{ik} = \frac{\pi_k g_k(\mathbf{x}_i, \theta_k)}{\sum_{k=1}^{K} \pi_k g_k(\mathbf{x}_i, \theta_k)}. \tag{11.133}$$
This is the E-step.
2. Update the mixing probabilities $\pi_k^{new} = \sum_{i=1}^{N} \tau_{k,i}$, for each component $k = 1, \ldots, K$.
3. Update the mean for each component $k = 1, \ldots, K$,
$$\boldsymbol{\mu}_k = \frac{\sum_{i=1}^{N} \tau_{k,i} \mathbf{x}_i}{\sum_{i=1}^{N} \tau_{k,i}}. \tag{11.134}$$
Note this is a weighted average, weighted based on the membership probabilities.

4. Update the Covariance matrix for each component $k = 1, \ldots, K$,

$$\boldsymbol{\Sigma}_k = \frac{\sum_{i=1}^{N} \tau_{k,i}(\mathbf{x}_i - \boldsymbol{\mu}_k)(\mathbf{x}_i - \boldsymbol{\mu}_k)^T}{\sum_{i=1}^{N} \tau_{k,i}}. \tag{11.135}$$

I.e. a weighted version of the sample covariance.

The number of groups, $K$, is a hyperparameter that must be chosen. Although this could be picked during cross-validation, a likelihood function framing means that information criteria (as introduced in Sect. 8.2.2) can also be used to find the most appropriate number of clusters. For different sized clusters calculate the BIC (or AIC). Plotting the BIC against the number of clusters can be used to find the point where increasing the number of clusters shows diminishing returns in terms of the drop in the BIC. This point is the "elbow" point of the plot (and hence why this heuristic is called the"elbow method") and indicates one choice for a suitable number of clusters. "Suitable" here is a relatively subjective term since there may be several other reasons why different numbers may be more appropriate or useful.

An example of the method is illustrated in Fig. 11.5. Here, the optimal number of clusters is around four since the tangential lines intersect around this value. Tangential lines are often used to make it easier to identify the elbow and hence the number of clusters.

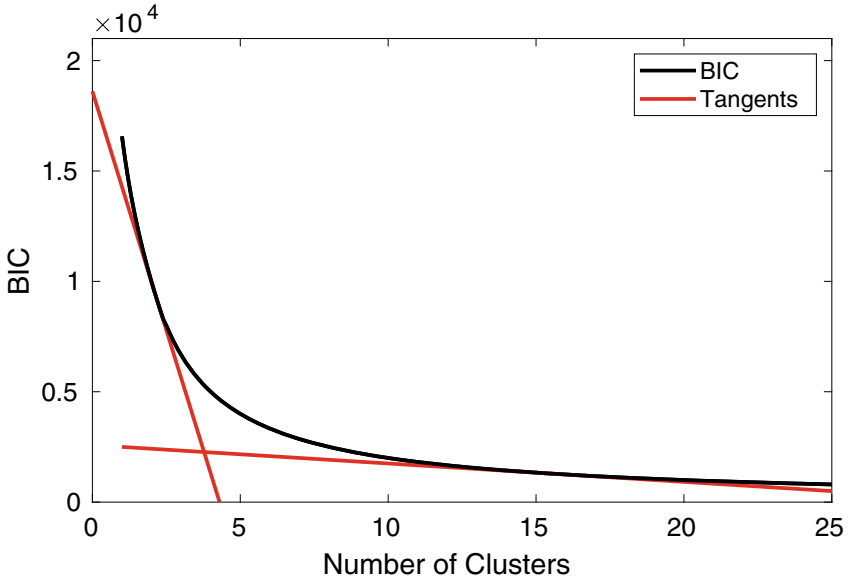


**Fig. 11.5** Example of Bayesian information criteria for different numbers of clusters in a GMM. Also shown are tangents to the curves to demonstrate the 'elbow plot' method for determining the 'optimal' number of clusters

Fitting a GMM is an easy way to estimate a distribution *if* all the data comes from the same distribution. For a time series this means that the data used for training forms a stationary series. Unfortunately this is unlikely to be the case in general. Different hours of the day may have different distributions and the time series may be dependent on weather, time of year, or a whole host of other variables. Hence, even though the EM-algorithm allows for relatively quick training of the GMM, there may be insufficient data to train several mixture models accurately.

## 11.4 Quantile Regression and Estimation

The majority of models used for probabilistic load forecasting are nonparametric and are popular because they allow more flexibility in what distributions are being modelled. One of the simplest and most common ways to generate univariate probabilistic forecasts is quantile regression, the subject of this section. One of the advantages of the method is that it is a simple adaption of standard least squares regression.

Consider estimating the $q$ quantiles (See Sect. 3.2 for introduction to quantiles) for the time steps $t = N + 1, N + 2, \ldots, N + k$. Popular choices are deciles (10-quantiles) or demi-deciles (20-quartiles) so that the distribution is split into 10 or 20 areas of equal probability respectively.

Consider the historical time series $L_1, L_2, \ldots, L_N$. Recall from Sect. 9.3, the aim in standard linear regression is to find the parameters $\boldsymbol{\beta}$ of some forecast model $f_t(\mathbf{Z}, \boldsymbol{\beta})$ by minimising the least squares difference with the observations. In mathematical terms this can be written

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta} \in \mathcal{B}} \left( \sum_{t=1}^{N} (L_t - f_t(\mathbf{Z}, \boldsymbol{\beta}))^2 \right). \tag{11.136}$$

Here $\mathcal{B}$ represents the set of feasible values the parameters can take, this is often the multi-dimensional real space $\mathbb{R}^p$, where $p$ is the number of parameters for the chosen forecast model. Once the parameters are found the model can then be used to produce forecast values using new inputs.

For quantile regression the principle is identical, except now instead of minimising a least squares cost function, for each quantile $\tau \in \{1, 2, \ldots, q\}$ a set of parameters, $\hat{\boldsymbol{\beta}}_\tau$ must be found which minimise the difference between the model and the observations according to the quantile loss function, i.e.

$$\hat{\boldsymbol{\beta}}_\tau = \arg\min_{\boldsymbol{\beta} \in \mathcal{B}} \left( \min \sum_{t=1}^{N} c_\tau(L_t, f_t(\mathbf{Z}, \boldsymbol{\beta})) \right), \tag{11.137}$$

where the cost function $c_\tau(x, y)$ is defined by

$$c_\tau(x, y) = \begin{cases} \tau(x - y) & x \geq y \\ (1 - \tau)(y - x) & x < y \end{cases},$$

This is repeated for all quantiles. Recall this is the same pinball loss score introduced in Chap. 7. The process of quantile regression is slightly more complicated than for least squares regression as the cost function isn't differentiable. However the problem is easily reformulated as a linear programming problem and can be solved very quickly. Quantile regression is only applicable for models, $f_t(\mathbf{Z}, \beta)$, which are linear combination of the parameters. This still allows a lot of versatility in the types of relationships that can be modelled.

To illustrate the process consider a very simple example. Generate 400 points from a Gaussian distribution (See Sect. 3.1) with mean $\mu = 2$ and standard deviation $\sigma = 3$ to represent a time series of 400 points. Here the $y$-axis values are the random points and the order in time is simply the order in which they were sampled. The time series is shown in black in Fig. 11.6. Now consider a simple linear model of the form $f_t(\boldsymbol{\beta}) = at + b$, (i.e. the parameters $\boldsymbol{\beta} = (a, b)^T$) where there is no other inputs $\mathbf{Z}$ since there is no dependencies in this particular model. A quantile regression for this linear model is applied to the time series as in Eq. (11.137) for each deciles (or 10-quantiles). These are shown in Fig. 11.6 as the red dashed lines. Notice in theory, in the limit of increasing numbers of samples, the final quantiles should be flat horizontal lines, and should describe quantiles for a Gaussian distribution with mean

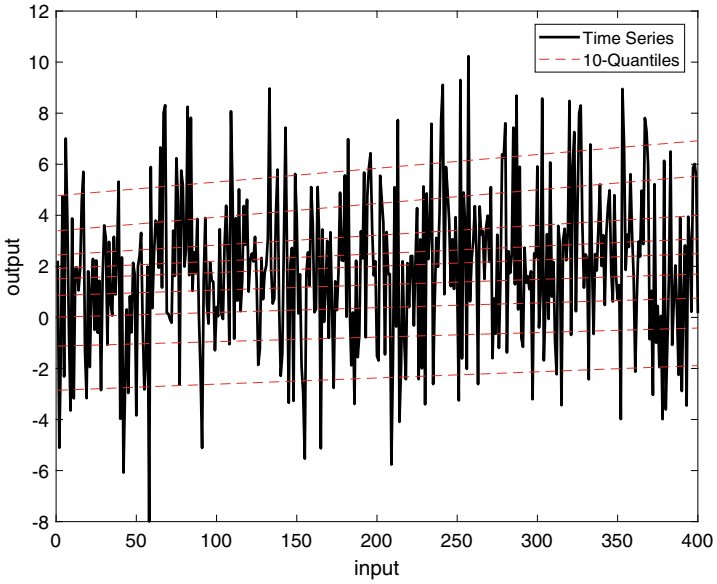


**Fig. 11.6** Random time series (black) and the 10-quantiles generated from a quantile regression applied to the simple linear model $a + bt$

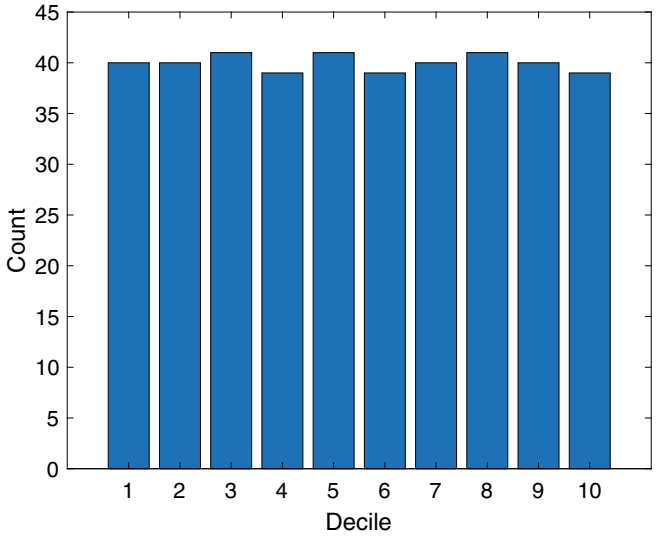


**Fig. 11.7** Plot of the probability integral transform for the example in the text. This shows the count of observations in each decile as defined by the quantile regression on the linear model to the data

$\mu = 2$ and standard deviation $\sigma = 3$. However, in this case there is a slight gradient since there is only a relatively small amount of data and skews in the sampling can have a large effect on the model fit.

Recall in Chap. 7 that the Probability Integral Transform (PIT) can be used to assess the calibration of a probabilistic forecast. The quantile regression lines should split the data into equal probabilities of observations which would mean 40 (400/10) observations are expected between each of the consecutive deciles. This is shown to be the case in the PIT in Fig. 11.7. Notice in some quantiles there is actually 39 or 41 observations due to the relatively small number of samples. Having a uniform PIT on the training set should be expected when an appropriate model is chosen. The true assessment of the model is, as always, determined by evaluating it on an unseen test set rather than the training set. In addition, for a probabilistic forecast both calibration and sharpness are important properties and therefore the proper scoring functions introduced in Sect. 7.2 should be used to evaluate forecasts rather than the PIT alone.

Finally, it should be noted that since each quantile is trained independently, sometimes the quantiles may cross over with each other which would obviously be inconsistent since say the 5% quantile may end up higher than the 10% quantile. To prevent this, it is valid to reorder each quantile at each time step to ensure the $p$ percentile is lower than the $q$ percentile, when $p < q$.

## 11.5 Kernel Density Estimation Methods

As shown in Sect. 3.4 kernel density estimation (KDE) could be viewed as a smooth version of a histogram. However, instead of adding discrete counts in different buckets, a continuous distribution can be estimated by adding kernel functions at the positions where observations are made. Consider observations $X_j$, for $j = 1, \ldots, N$ of a random variable $X$, then the KDE for the probability density function is defined as

$$\hat{F}(X) = \frac{1}{Nh} \sum_{j=1}^{N} K\left(\frac{X - X_j}{h}\right), \tag{11.138}$$

where $h$ is the bandwidth, a smoothing parameter for the estimate, and $K()$ is some kernel function. A popular example of the kernel function is the so-called Gaussian kernel defined as

$$K(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}x\right). \tag{11.139}$$

The chosen kernel is often less important than the proper training of the bandwidth. Also note that the choice of kernel function has no relationship to the true distribution, i.e. a Gaussian kernel does not mean the data is distributed as a Gaussian. The importance of the bandwidth is illustrated in Fig. 11.8. The plot shows a comparison between the histogram of the 200 observations (left) versus the KDE of the same observations but for three different bandwidths (right). Selecting a bandwidth too small and the KDE will overfit the observations, too large and the KDE will underfit and have a higher bias (recall Sect. 8.1.2 on bias-variance tradeoff). Although

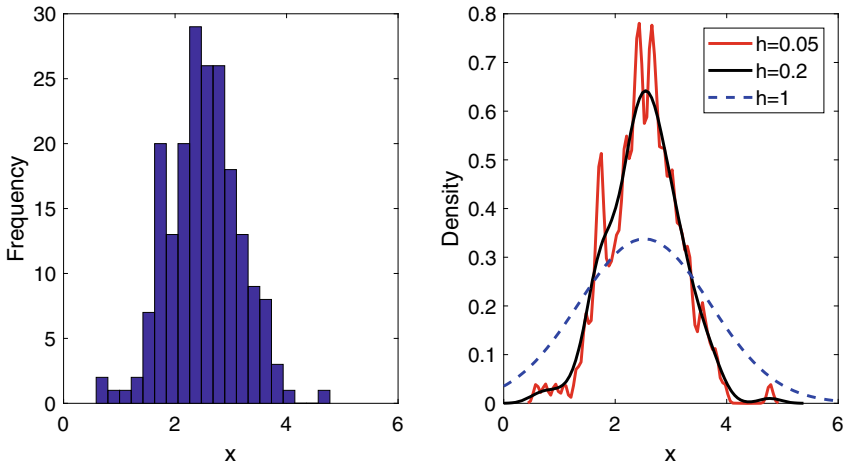


**Fig. 11.8** Examples of estimating a distribution from 200 observations using **a** a histogram with 20 equally spaced bins and **b** a kernel density estimate with different bandwidths

there are rules of thumb for choosing the bandwidth, these are often based on strong assumptions of the underlying distribution (such as being Gaussian) and hence are too restrictive for the purposes of load forecasting. Instead the bandwidth can be chosen using cross validation by minimising the fit (often defined according to minimising a probabilistic scoring function such as the CRPS, see Chap. 7) between the estimate and the observations in the validation set. This could be through a search of the hyperparameter space e.g. grid search (Sect. 8.2.3). Although there is only one parameter for the simplest form of KDE, it can be a computationally expensive process.

The KDE can be easily adapted to probabilistic time series forecasts. Consider again historical observations $L_t$ for $t = 1, \ldots, N$ for some random variable which are assumed to come from the same distribution with the aim being to generate k-step ahead density forecasts for each time step $t = N + 1, \ldots, N + k$ in the forecast horizon. In this simplified case the most basic kernel density estimate can be defined as

$$\hat{F}_i(L_{N+i}) = \frac{1}{Nh} \sum_{t=1}^{N} K\left(\frac{L - L_t}{h}\right), \tag{11.140}$$

for each time step in the horizon $i = 1, \ldots, k$, and bandwidth $h$. In other words, the distribution is assumed to be the same at each time step. This is clearly unrealistic for several reasons. For one, it is likely that older data is less relevant than more recent information. In addition, the KDE estimate in Eq. (11.140) is also independent of any other inputs, e.g. temperature or time of the day/week. To rectify these shortcomings modified versions of the simple KDE estimate are available.

To reduce the influence of older points a simple decay factor, $\lambda$ with $0 < \lambda \leq 1$, can be introduced. This reduces the contribution of older data to the overall distribution function. One possible implementation is

$$\hat{F}_i(L_{N+i}) = \sum_{t=1}^{N} w_t K\left(\frac{L - L_t}{h}\right), \tag{11.141}$$

where the exponential decay weight $w_t = \frac{\lambda^{N-t}}{h \sum_{l=1}^{N} \lambda^{N-l}}$. In this case both the decay factor $\lambda$ and the bandwidth must be optimised (again, often by cross-validation). Other weightings are of course possible, as will be seen with the conditional KDE form below. If more historical data is likely to be relevant then a slower decay, e.g linear, may be of interest. The only restriction is that the weights should sum to one to ensure the final function is still a well-defined probability distribution.

Another simple modification is to train only on specific historical points. For example, load data often has simple seasonalities (such as daily or weekly) of integer period $s$. In this case, the density can be estimated using

$$\hat{F}_i(L_{N+i}) = \frac{1}{N_s h} \sum_{t \in \mathcal{I}_i} K\left(\frac{L - L_t}{h}\right), \tag{11.142}$$

where $N_s$ is the number of elements of the set $\mathcal{I}_i = \{j | N + i - j = sk$ for some integer $k\}$. $\mathcal{I}_i$ is every index which is a constant multiple of the period $s$, so in the case of hourly data with daily seasonality, to predict the $i = 2$ time period (e.g. 2AM if the forecast origin is at midnight) the historical data used to construct the KDE estimate would only use the data from the 2AM on each of the historical days. Recall the periodicities and seasonalities can be found using the methods presented in Chap. 5.

Another popular update to the simple KDE is conditional kernel density estimation. This estimates the distribution of the variable $L_i$, *conditional* on some dependent variables, say $T$, $S$. We can now utilise the pairs of independent-dependent observations as $(T_t, S_t, L_t)$ and define the conditional distribution of $\hat{F}_i(L_i|T, S)$ as

$$\hat{F}_i(L_{N+i}|T, S) = \sum_{t=1}^{N} \frac{K((T_t - T)/h_T)K((S_t - S)/h_S)}{\sum_{l=1}^{n} K((T_l - T)/h_T)K((S_l - S)/h_S)} K\left(\frac{L - L_t}{h}\right), \tag{11.143}$$

were $h_T$, $h_S$ are the bandwidths for the distributions representing $T$ and $S$ respectively and now must be found together with the dependent series bandwidth $h$. This is, as before, simply a weighted sum like in Eq. (11.141), but where the weights are kernel based functions of the dependent variables. As usual, popular choices of the independent variables are weather variables, but also period of the week. This can also be extended or simplified to take into account less or more variables respectively. However each bandwidth significantly increases the computational cost of training the models which can be impractical beyond two conditional independent variables.

Often to help accelerate the optimisation, the variables are normalised (e.g. to $[0, 1]$) in order to reduce the search space (see Sect. 6.1.3). The forecast can be rescaled after the training is complete. As mentioned in Sect. 3.4 there are options for the different kernels, and different ones can be tested, although often the choice has minimal impact on the accuracy of the forecasts [1].

The different modifications presented here can obviously be combined to create other models. For example the conditional kernel density form shown in Eq. (11.143) can be extended to include a decay factor like in Eq. (11.141) or restrictions can be applied on the inputs like in Eq. (11.142). As with many KDE methods, the drawback is that each modification often increases the training complexity and computational cost.

## 11.6 Ensemble Methods

This section introduces **ensemble forecasts**, by which we mean a set of point forecasts from the same forecast origin, estimating each time step with the same forecast horizon (of length $h$ time steps). The point forecasts are samples of equal probability from a $h$-dimensional multivariate distribution representing the joint distribution over the forecast horizon (see Sect. 3.3 for more on joint distributions). In other words,

each ensemble represents an equally likely load trajectory. The methods described in this section produce these ensembles without needing to produce the full joint distribution.

### *11.6.1 Residual Bootstrap Ensembles (Homoscedasticity)*

This section describes a method for generating ensemble forecast which can be viewed as realisations from a full multivariate probabilistic forecast in the case where the time series is assumed to have fixed variance. To begin consider a 1-step ahead point forecast model, for example, this could be the exponential smoothing model in Sect. 9.2 or the ARIMA models in Sect. 9.4. Denote this as $f(L_t|\mathbf{Z}, \boldsymbol{\beta})$ which may use previous historical data as well as any explanatory inputs (all described by the set of variables $\mathbf{Z}$) to create an estimate, $\hat{L}_{t+1|t}$, for the true value, $L_{t+1}$, at $t+1$. The $\boldsymbol{\beta}$ are the parameters for the model. The next time step ahead can be forecast by iteratively applying the model and including the forecast from the previous time step as a pseudo-historical input to the model. Hence for the next time step

$$\hat{L}_{t+2|t} = f(\hat{L}_{t+1|t}|L_t, \mathbf{Z}, \beta) = f(f(L_t|\mathbf{Z}, \beta)|L_t, \mathbf{Z}, \beta). \tag{11.144}$$

The process can obviously be repeated to produce $k$-step ahead forecasts. Now, recall that there is an error process describing the difference between the observations and the 1-step ahead forecasts described by the residual $\epsilon_{t+1} = L_{t+1} - \hat{L}_{t+1|t}$. By including the small deviations, described by the residual series, into the forecast, different trajectories can be created which represent different, but equally likely outcomes.

To describe the algorithm in more detail, consider a $k$-step ahead forecast generated from a one step ahead model, e.g. $\hat{L}_{t+1} = f(L_t|\mathbf{Z}, \boldsymbol{\beta})$. Assume the forecast origin is at time step $t = N$. Hence the aim is to produce ensemble forecasts which cover the period $N+1, \ldots, N+k$. The residual series, $\epsilon_t = L_{t+1} - \hat{L}_{t+1}$ (which is calculated for the entire training set), is assumed to have a fixed variance (the series is said to have **homoscedasticity**) and are uncorrelated with each other. Using this residual series the process of generating a new ensemble for a $k$-step ahead forecast, using a 1-step ahead forecast model $f$, is relatively simple. For each ensemble, $b$, the procedure is as follows:

1. Randomly sample with replacement (this is called a bootstrap sample) a residual, $\hat{e}_1^{(b)}$, from the set of all residuals, $\{\epsilon_1, \epsilon_2, \ldots, \epsilon_N\}$.
2. Add this residual to the current 1-step ahead forecast value $\tilde{L}_{N+1|N}$ to produce a new value $\hat{L}_{N+1|N}^{(b)} = \tilde{L}_{N+1|N} + \hat{e}_1^{(b)}$.
3. Include $\hat{L}_{N+1|N}^{(b)}$ in the forecast model to generate an estimate for the next time step, $\tilde{L}_{N+2|N+1}^{(b)} = f(\hat{L}_{N+1|N}^{(b)}|L_N, \mathbf{Z}, \beta)$.

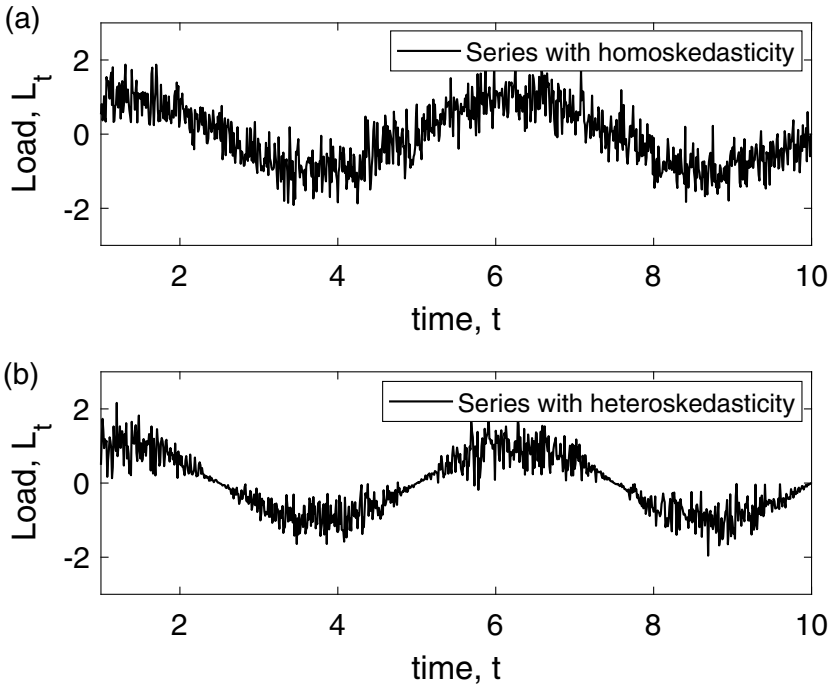


**Fig. 11.9** Example of a simple periodic time series with homoscedasticity (top) and heteroskedasticity (bottom)

4. Update this value using another bootstrap sample from the residual series to give $\hat{L}^{(b)}_{N+2|N+1} = \tilde{L}^{(b)}_{N+2|N+1} + \hat{e}^{(b)}_2$.
5. Continue this procedure until the $k$th step is reached.
6. The final series, $\hat{L}^{(b)}_{N+1|N}, \hat{L}^{(b)}_{N+2|N+1}, \ldots, \hat{L}^{(b)}_{N+k|N+k-1}$ is the $b$th bootstrap ensemble.

This is also known as a **residual bootstrap forecast**. The process can be repeated to produce as many ensembles as desired. The more ensembles generated, the more load diversity is captured. Generating more samples increases the computational cost, but since each ensemble is independent of the others they can be generated in parallel. Notice that this method strongly assumes that the 1-step ahead errors in the future will be similar to the past 1-step ahead errors. An example of a simple periodic series with homoscedasticity is shown in Fig. 11.9a.

If instead of sampling from the actual residuals you sample from an assumed or fitted distribution then the method can be referred to as a Monte Carlo forecast. For example, it is often assumed that residuals are Gaussian distributed with zero mean and therefore instead of sampling from the set of residuals, the values can be sampled from a Gaussian distribution trained on the residuals. An example of an ensemble forecast generated from the Monte Carlo simulations, for 100 ensembles is shown

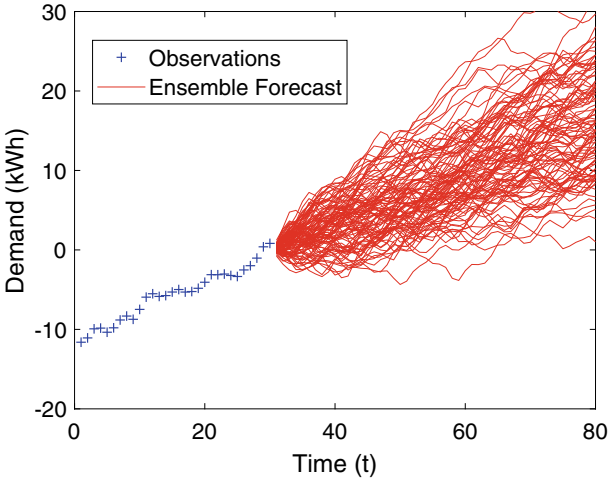


**Fig. 11.10** Example of an Monte Carlo derived ensemble 50-step ahead forecast with 100 ensembles for a simple ARIMA model

in Fig. 11.10 for a simple ARIMA(4, 1, 1) model. Notice that the errors get wider (have larger variation) with forecast horizon length. This is due to the accumulation of the errors from one step to the next. This intuitively makes sense as the uncertainty should increase the further ahead the prediction.

Notice that the forecasts at each time step can be used to estimate a univariate estimate. This can be done by either fitting quantiles or a density estimate to the collection of ensemble points at each time step.

### *11.6.2 Residual Bootstrap Ensembles (Heteroskedasticity)*

An advantage of the bootstrap method described in Sect. 11.6.1, is that a multivariate forecast can be generated with minimal computational cost since only the original point forecast model needs to be trained. Further, if the model contains autoregressive features (as ARIMA and exponential smoothing do) then the ensembles also retain the interdependencies of the time series. A drawback to the method is the strong assumption of homoscedasticity for the series of residuals. In fact, it is likely that periods of high demand will also have larger variability. A time series where the variance changes in time is said to have **heteroskedasticity**.

An example of a simple periodic series with heteroskedasticity is shown in Fig. 11.9b in which the largest variation in the demand coincides with the largest amplitude of the periods. When time series are heteroscedastic the variability can be incorporated using so-called GARCH-type models which can extend the bootstrapping method described in Sect. 11.6.1. An outline of these methods are given here,

but since they are relatively complicated, the details are ommited. The interested reader is referred to some further reading given in Appendix D.

Consider a model for the load time series with some mean process (e.g. any of the point forecasts presented in Chaps. 9 and 10) and as before let $\epsilon_t$ be the 1-step ahead error/residual terms. Now assume that the standard deviation $\sigma_t$ also varies with time $t$. It is helpful to write the residual in the form

$$\epsilon_t = \sigma_t Z_t, \tag{11.145}$$

where $\sigma_t > 0$ is a time-dependent conditional standard deviation, and the $(Z_t)_{t_\epsilon \mathbb{Z}}$ is a random variable which is stationary, independent in time and has the conditions $\mathbb{E}(Z_t) = 0$ with $\mathbb{V}ar(Z_t) = 1$. Splitting the data into the standard deviation this way helps to simplify the actual variation into the magnitude of variation, represented by the standard deviation, and the random component which is now stationary due to the scaling. Once the components of Eq. (11.145) are found it is easy to apply an adapted form of the bootstrap method by taking random samples from the distribution of $Z_t$, and then rescale with the modelled standard deviation $\sigma_{N+k}$ at the time step being forecast.

To do this, first a model must be chosen for the standard deviation. In economic models standard choices are ARCH and GARCH models. The ARCH and GARCH models are essentially variance counterparts to the AR and ARMA models of the point forecasts introduced in Sect. 9.4. The GARCH(p, q) model is of the form

$$\sigma_t^2 = \alpha_0 + \sum_{i=1}^{q} \alpha_i \epsilon_{t-i}^2 + \sum_{j=1}^{p} \beta_j \sigma_{t-j}^2, \tag{11.146}$$

where the $q$ is the lag residual terms (called the ARCH term) and $p$ is the lag variance terms (this is called the GARCH term). The ARCH model only has the $q$ lags with the residual terms and no variance terms. These are often coupled with the corresponding AR, or ARIMA model for the mean process and are solved with ordinary least squares or maximum likelihood methods.

ARCH and GARCH are specific forms of the variance which are often suitable for financial time series applications. In fact standard deviation can be modelled much more generally and these will be referred to as GARCH-type methods. In load forecasting, the variation in the demand is often larger for time periods when the demand is typically higher (however, of course, for each new time series the patterns in the variance should be analysed before choosing a model). For this reason it is often suitable to choose a standard deviation model which is similar to the point forecast model chosen.

For ARCH and GARCH problems the procedures are relatively well established and hence there are many packages for automatically selecting the order and coefficients for the model. In the more general case a simple procedure can be followed which is adapted from [2]:

1. Since the components of the residuals as given in Eq. (11.145) are not known, instead consider the absolute or squared residuals $|\epsilon_t|$ or $\epsilon_t^2$. The former will be the focus as it better captures the heavy tails of energy demand. However, the same procedure is applicable for both forms.
2. Since $\mathbb{E}(|\epsilon_t|) = \sigma_t \mathbb{E}(|Z_t|))$, fitting a model to $|\epsilon_t|$ is equivalent to fitting a model to a scaled version of the standard deviation, $C\sigma_t$, for some constant $C > 0$ since $Z_t$ is a stationary variable (and hence has constant expectation). The fit is usually achieved using ordinary least squares (Sect. 8.2).
3. The constant $C$ must now be estimated by considering the normalised residuals, $\epsilon_t/|\epsilon_t| = \epsilon_t/C\sigma_t = Z_t/C$, i.e. a scaled version of $Z_t$. Since $Z_t$ has variance equal to one, the scaling, $C$, can be be estimated by calculating the sample standard deviation, $\alpha$, of the normalised residuals which tells us that $C = 1/\sqrt{\alpha}$.

For this method notice that there is no assumption on the underlying distribution (only on its variance). An updated version of the bootstrap forecast in Sect. 11.6.1 now updates the 1-step ahead forecast by adding a sample from the distribution of $Z$ (sample from the empirical distribution for $Z$ formed from the standardized residuals $\epsilon_t/\sigma_t$) which has then been scaled by the $\sigma_t$ model at the current time. Note it is assumed that the errors $\epsilon_t$ and standardised residuals are uncorrelated in time to allow them to be randomly sampled for the bootstrap. As usual residuals should be checked (in this case the standardised residuals) to ensure that they satisfy the assumptions about correlation, fixed variance and have zero mean. If these assumptions do not hold then further updates can be applied (as in Sect. 7.5). A specific example of the above GARCH-type model will be given in the LV case study in Sect. 14.2.

## 11.7 Copula Models for Multivariate Forecasts

This section introduces **Copulas**, another popular method for generating multivariate probabilistic forecasts, widely used in quantitative finance, but now used extensively in energy forecasts. Here only the basics will be described. Suggested further reading can be found in Appendix D.

A copula is simply a function, $C$, from an $N$-dimensional unit box to a 1-dimensional unit box, i.e. $C : [0, 1]^N \longrightarrow [0, 1]$, and describes a cumulative distribution function on variables $U_1, \ldots, U_N$ where each variable $U_i$ is uniformly distribution over $[0, 1]$, i.e. $C(u_1, u_2, \ldots, u_N) = P(U_1 \leq u_1, U_2 \leq u_2, \ldots, U_N \leq u_N)$. In other words a copula models the joint distribution on variables $U_1, \ldots, U_N$ with uniform marginal distributions (See Sect. 3.3 for more details on joint and marginal distributions). A copula is focused on the correlation/inter-dependence structure of the variables. An advantage of these methods is the wide range of different possible copula functions which can be used to model the inter-dependence.

The power of copulas lie in the fact that, due to *Sklar's theorem*, any multivariate distributions can be modelled using only the marginal distributions and a copula. Consider a random variable $\mathbf{X} = (X_1, X_2, \ldots, X_N)^T \in \mathbb{R}^N$ with joint distribution

$F_{X_1,\ldots,X_N}(x_1,\ldots,x_N)$ and marginal CDFs denoted by $F_1(x_1),\ldots,F_N(x_N)$, then by Sklar's theorem, there exists a copula $C$ such that

$$F_{X_1,\ldots,X_N}(x_1,\ldots,x_N) = C(F_1(x_1),\ldots,F_n(x_N)). \tag{11.147}$$

Note that for any random variable $X$ with CDF $F$, is uniformly distributed when transformed by its CDF, i.e. $F_X(X)$ is uniformly distributed. Recall this is just the probability integral transform (PIT) described in Chap. 7.

The process in Eq. (11.147) is reversible which means once a copula model has been trained on observations then multivariate samples can be easily produced with the relevant dependency structure. First samples $u_1, u_2, \ldots, u_N$ are generated from the copula distribution and then each variable is transformed into the original random variable space using the inverse CDF of the marginal $F_i^{-1}(u_i)$ for each corresponding component of the sample point.

Suppose that $\mathbf{X} = (X_1, X_2, \ldots, X_N)^T \in \mathbb{R}^N$ has a multivariate Gaussian distribution, then the corresponding copula is defined purely by the correlation matrix since this explains the entire dependency structure. This also means that each correlation matrix defines a specific Gaussian copula. Of course just because a multivariate random variable has a Gaussian copula doesn't mean that it follows a multivariate Gaussian distribution since the marginals need not be Gaussian. If the correlation matrix is the identity then the copula is called the **independence copula** defined by

$$C_0(u_1, u_2, \ldots, u_N) = u_1 \ldots u_N, \tag{11.148}$$

where each component is independent of the other components. In general, there is no simple analytic form for a **Gaussian copula** but it can be expressed as

$$C_R^{Gauss}(u_1, u_2, \ldots, u_N) = \Psi_R(\Psi^{-1}(u_1), \ldots, \Psi^{-1}(u_N)), \tag{11.149}$$

where $\Psi$ is the univariate CDF of the standard Gaussian (mean zero and unit standard deviation) and $\Psi_R$ is the multivariate Gaussian with zero mean and correlation matrix $R$.

Another family of popular copula's are the **Archimedean copulas**, which have explicit formula's and can represent multivariate distributions using only one parameter, $\theta$. They have the general form

$$C(u_1, \ldots, u_N; \theta) = \psi^{-1}\left(\psi(u_1; \theta) + \cdots + \psi(u_N; \theta); \theta\right) \tag{11.150}$$

where $\psi : [0, 1] \times \Theta \to [0, \infty)$ is a continuous, strictly decreasing, convex function such that $\psi(1; \theta) = 0$. For example one popular Archimedean copula is the Gumbel Copula, which is defined by

$$C_{Gum}(u_1, u_2, \ldots, u_N|\theta) = \exp\left[-\left((-\log(u_1))^\theta + \cdots + (-\log(u_N))^\theta\right)^{1/\theta}\right]. \tag{11.151}$$

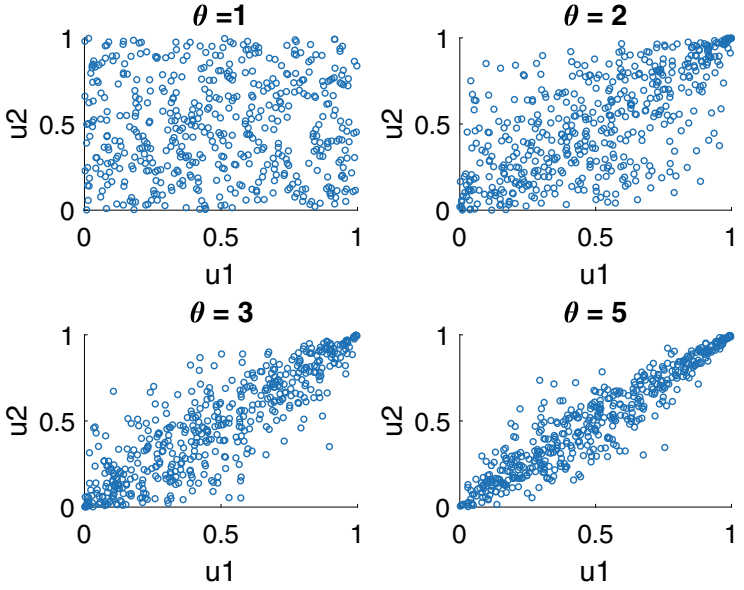


**Fig. 11.11** Example of samples from Gumbel copulas with different values of $\theta$

Notice that this becomes the independence copula when $\theta = 1$. Examples of samples from a bivariate Gumbel copula with different values of $\theta$ are shown in Fig. 11.11. Note that the Gumbel can never represent negative correlation.

It is clear that different copula's are useful for different dependency structures and that not all copulas are useful for all types of data. For example, Gumbel copula's shouldn't be used with data with negative correlations. How to choose and fit a copula will be briefly considered later in this section.

The value of the Pearson's correlation coefficient depends on the marginals and the copula, which means random variables will have different Pearson values when transformed using the marginal CDFs. A more convenient measure for the correlation structure for a copula are Rank correlation coefficients which only depend on the *rank* of the data (see Sect. 3.5). Since the rank of data is unchanged by the application of a monotonically increasing function, it means the rank correlation coefficients won't change when the marginal CDFs are applied. An example of a rank correlation coefficient was given in Sect. 3.5: the Spearman's rank correlation coefficient.

To better understand how copula's work and how they can be used to generate new samples, consider a simple bivariate distribution for the random variables $(X1, X2)$, whose dependency structure is described by a Gaussian copula with covariance

$$R = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$$

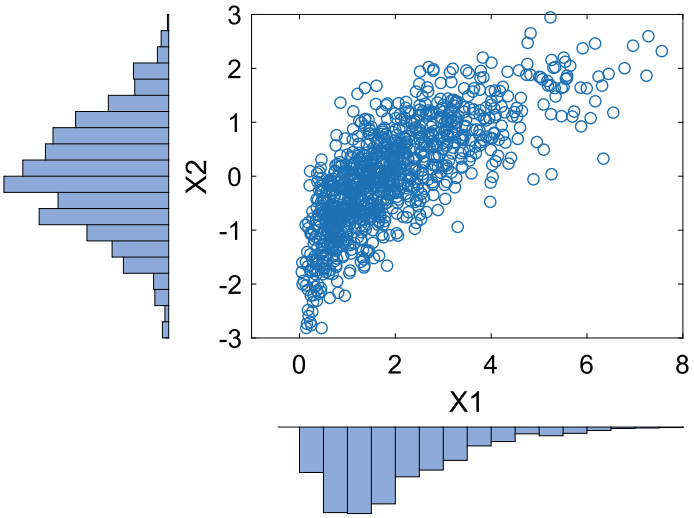


**Fig. 11.12** Observations from a bivariate distribution with Gaussian Copula, with Gamma ($X1$) and Gaussian ($X2$) marginal distributions (shown as the histograms)

i.e. it has linear Pearson correlation $\rho = 0.8$. Also suppose that the marginal of the first variable, $X1$, has a distribution described by a Gamma distribution, where

$$Gamma(X, \alpha, \beta) = \frac{1}{\beta^{\alpha}\Gamma(\alpha)} \int_0^X t^{\alpha-1} e^{-t/\beta} dt \tag{11.152}$$

where $\alpha = 2$ is the shape parameter (determines the shape of the distribution), and $\beta = 1$ is the scale parameter (determines how spread the distribution is). $\Gamma(.)$ is the so-called gamma function. The second variable, $X2$, is described by the standard Gaussian distribution (mean zero and unit standard deviation). An example of 1000 observations from this distribution is shown in Fig. 11.12. On the horizontal and vertical of the plots are marginal histograms of each variable, which shows the Gamma and Gaussian distributions respectively.

The distribution after applying the CDF of each marginal to its respective variable is shown in Fig. 11.13. The marginals are now described by uniform distributions, as expected. Notice in this example the sample pearson correlation between $X1$ and $X2$ is $\rho = 0.767$ but between $U1$ and $U2$ is $\rho = 0.785$ and is not preserved by the transformation (although they are close in this example). In contrast the Spearman's correlation between $X1$ and $X2$, and between $U1$ and $U2$ are both $r = 0.787$ as expected. In practice the objective would be to fit a copula to this transformed data. In this case assume it is known that the copula is a Gaussian and therefore the aim is to find the Pearson correlation coefficient $\rho$. In fact in the [3] bivariate case the Spearman's coefficient $r$ is related to the linear correlation via

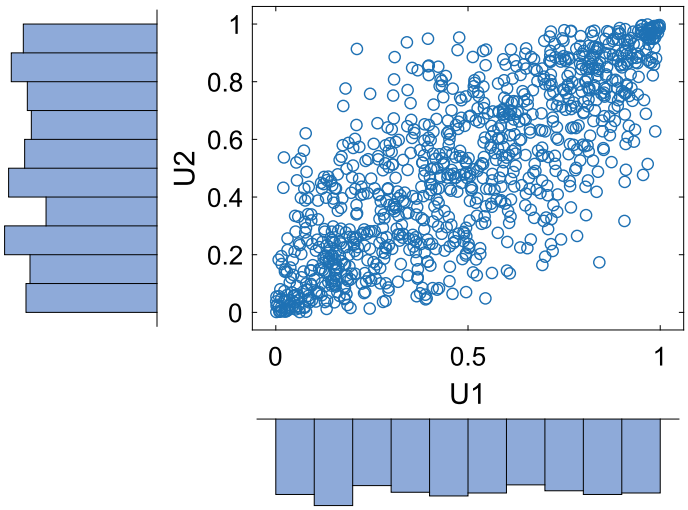


**Fig. 11.13** Distribution of transformed observations where the marginals of their respective components have been applied. The marginals are now uniform as shown by the histograms

$$\rho = 2 \sin\left(r \frac{\pi}{6}\right), \tag{11.153}$$

and hence the invariant value of the Spearman's correlation can be used to estimate the Pearson correlation and gives $\rho = 0.801$ (note it isn't exactly the 0.8 used to generate the data due to numerical deviations in the sample, the larger the sample the closer we would expect the sample value to be to the original parameter).

Given the copula, new samples can be generated and then transformed to the original space (with the same linear correlation) by using the inverse CDF of the marginals. An example of 1000 new points generated using the copula (and transformed using the inverse CDFs) is shown in Fig. 11.14. Notice how the final distribution successfully resembles the original distribution in Fig. 11.12.

A similar process can be used to generate ensemble demand forecasts. In this case, consider a demand time series $L_1, L_2, \ldots$, where the aim is to generate a day-ahead multivariate forecast with, say, forecast origin $t = N$. Further for simplicity, suppose the data is hourly and hence a 24-step ahead forecast is being considered. The aim here will be to generate a multivariate distribution for the day, $F_{N+1,\ldots,N+24}(L_{N+1}, \ldots, L_{N+24})$, and hence model the inter-dependencies between different times of the day. It is assumed that CDFs for the marginals at different times of the day are already known, i.e. $F_1(L_{N+1}), \ldots, F_{24}(L_{N+24})$ are known. These could be estimated, e.g. by the univariate probabilistic models described earlier in this chapter. In this case a copula can be used to model the intra-day dependency structure by training on the daily profiles transformed by the marginals.

There is a range of different copula models, only some of which are mentioned above. The questions remain on how to train and choose the copulas on the data.

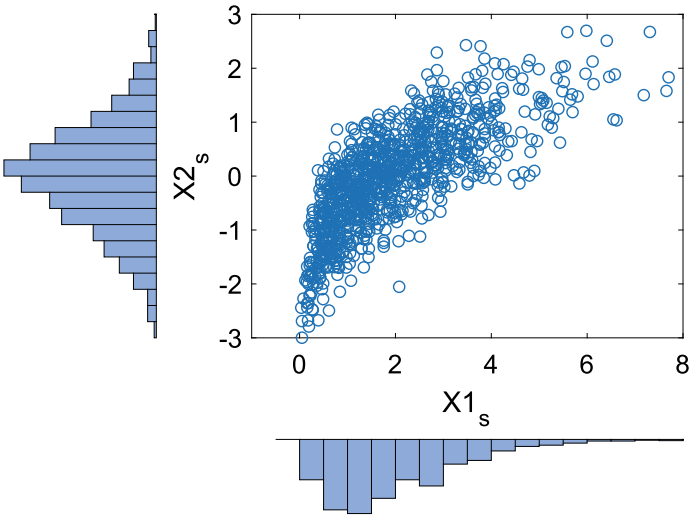


**Fig. 11.14** Samples from the copula model fitted to the observations

In theory, for parametric models for the copulas and marginals, maximum likelihood estimation (see Sect. 8.2) could be used to fit a copula, however this can be complicated for high dimensional problems as there are lots of parameters to train. Instead the models can be estimated using a two step process called the *Pseudo-Maximum Likelihood*, where first the marginals are estimated and a reduced form of the maximum likelihood, given by

$$\sum_{k=1}^{n} \log \left[ c\{\hat{F}_1(X_{1,k}), \ldots, \hat{F}_N(X_{N,k}) | \boldsymbol{\theta}_C\}) \right], \tag{11.154}$$

is maximised. Here $c$ is the copula density corresponding to the Copula CDF, $C$, and $\boldsymbol{\theta}_C$ represents the parameters for the copula model. The maximisation above can still be complicated especially for higher dimensional problems, depending on the copula model considered. One approach has already been suggested for the simple Gaussian copula example given above. The correlation matrix can be estimated by using the Spearman's correlation coefficients for each pair of variables and this can either be used as final correlation matrix or as an initial guess in the pseudo-maximum likelihood optimisation in Eq. (11.154).

Choosing the correct copula's depends on many factors and a detailed investigation is beyond the scope of this book. Further reading is suggested in Appendix D. In summary, the choice depends on the type of correlation being modelled as well as the dependencies within the tails/extremes of the distribution. One possible approach for choosing an appropriate copula(s) model can be based on comparison on a validation set as described in Sect. 8.1.3.

## 11.8 Questions

For the questions which require using real demand data, try using some of the data as listed in Appendix D.4. Preferably choose data with at least a year of hourly or half hourly data. In all the cases using this data, split into training, validation and testing in a 60, 20, 20% split (Sect. 8.1.3).

1. Sample 20 points from a 5-dimensional Gaussian distribution. Make sure that some of the variables are more correlated than others by manipulating the correlation between them in the covariance matrix. You can fix the variance of all the variables to one to make the model simpler. Now consider that each dimension of the Gaussian is a different time step in a time series of five points. Plot each sample to create samples like in Fig. 11.2. What can you see between the variables which are highly correlated? What if you change the variance for different variables, how does this change the ensemble plot?
2. Generate a quantile regression. Take your linear forecast model you generated in Sect. 9.7. Now fit to the training data a quantile regression for percentiles of $10, 20, 30, \ldots, 90$ using inbuilt packages such as quantreg in R.[3] Apply to the test set, and count how many values lie between each set of quantiles. Plot the probability integral transform. What shape is it? Is there a bias in the model? Is it under or over dispersed? What adjustments to the quantiles could help produce a uniform PIT?
3. To demand data with daily or weekly periodicity fit a kernel density estimate for each time step from each period in the seasonal cycle. For example, if the data is half hourly with daily seasonality then train 48 models for each half hour of the day. Fit the model by performing a grid search for the bandwidth. With the final model, apply it to the test set. Generate quantiles for the estimate, and thus calculate the PIT for the same percentiles as the previous question. Is the PIT uniform, overdispersed or underdispersed?

## References

1. J. Jeon, J.W. Taylor, Using conditional kernel density estimation for wind power density forecasting. J. Amer. Stat. Ass. **107**(497), 66–79 (2012)
2. S. Haben, G. Giasemidis, F. Ziel, S. Arora, Short term load forecasting and the effect of temperature at the low voltage level. Int. J. Forecast. **35**, 1469–1484 (2019)
3. T.C. Headrick, A note on the relationship between the pearson product-moment and the spearman rank-based coefficients of correlation. Open J. Stat. **6**, 1025–1027 (2016)

[3] https://cran.r-project.org/web/packages/quantreg/index.html.