STAT 626: Outline of Lecture ~~16~~ 13

Focus On Forecasting (§3.5)

1. **Treating TS Data as Independent Observations: $\bar{x}$ IS the Predictor**
2. **Forecasting An AR(1) Model: The Past is Discounted by Correlation**
3. **Reading Assignment: Examples 3.36, Random Walk, and 3.37, IMA (1,1) and EWMA: Holt-Winter's Method. Extremely important in applied forecasting.**
4. **Forecasting a One-sided MA(∞) Time Series; Infinite Past**
5. **Prediction Error Variance (P_{n+1}^n) and Forecast Interval:**

$$x_{n+1}^n \pm 1.96\sqrt{P_{n+1}^n}$$

6. **Example 3.24 Forecasting the Recruitment Series**
7. **Durbin-Levinson Algorithm: PACF** Missing Data: Interpolation
8. Michael Abramowicz (2008).**Predictocracy: Market Mechanisms for Public and Private Decision Making.**

   Predicting the future is serious business for virtually all public and private institutions, for they must often make important decisions based on such predictions. This visionary book explores how institutions from legislatures to corporations might improve their predictions and arrive at better decisions by means of prediction markets, a promising new tool with virtually unlimited potential applications.

9. James Surowiecki (2005). **The Wisdom of Crowds: Why the Many are Smarter than the Few**

Who said the following ?

"It is difficult to make predictions, especially about the future"

FORECASTING

How is the statistical or scientific forecast different from that of a **fortune teller or psychic?**

Given the time series data $x_1, \ldots, x_n$: What is a good way to forecast the next future value x_{n+1}?

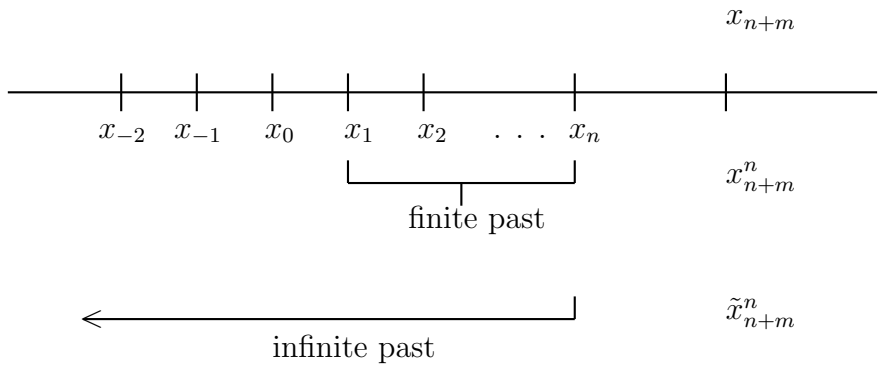
For the moment pretend the x_i are i.i.d. Then, their sample mean is the "best" predictor. Why?

In what sense is it the "best" predictor?

Forecasting a Stationary Series Based on the Infinite Past

Assumption: The ACF or the parameters, and the past are known.

A pictorial setup for forecasting the future values $x_{n+m}, m = 1, 2, \ldots$:



1. What are their forecasts, forecast error, and forecast error variances?

   Forecast error: $x_{n+m} - x^n_{n+m}$

   Error variance: $P^n_{n+m} = \mathrm{Var}(x_{n+m} - x^n_{n+m})$

2. Their 95% forecast intervals?

$$x^n_{n+m} \pm 1.96\sqrt{P^n_{n+m}}.$$

EXAMPLES:

1. Forecasting a Causal AR(1) Model

$$x_t = \phi x_{t-1} + w_t.$$

Its predictor?

Its prediction error variance?

Its 95% forecast interval?

2. Forecasting Causal ARMA Models:

3. One-Sided MA(∞) Processes:

$$x_t = \sum_{j=0}^{\infty} \psi_j w_{t-j},$$

with absolutely summable coefficients.

The forecast? The prediction error? Its variance?

4. Forecating ARIMA and SARIMA Models (Reading Assignment).

Forecasting Based on the Finite Past:

Regression Forecast: Given the value of a random variable X, **find β to minimize the mean-square error (MSE) of predicting Y by $\widehat{Y} = \beta X$:**

$$\mathrm{MSE}(\beta) = E(Y - \beta X)^2.$$

SOLUTION: The minimizer satisfies the *normal equation*:

$$\mathrm{Var}(X) \;\; \widehat{\beta} = \mathrm{Cov}(X, Y)$$

or

$$\widehat{\beta} = \frac{\mathrm{Cov}(X, Y)}{\mathrm{Var}(X)},$$

and

$$\mathrm{MSE}(\widehat{\beta}) = E(Y - \widehat{Y})^2 = (1 - \rho^2)\mathrm{Var}(Y).$$

Time Series Forecast: Given the time series data $x_1, \ldots, x_n$ from a zero-mean stationary process $\{x_t\}$ with **known** autocovariance function,$\gamma(h)$, find the forecast of the next future value, x_{n+1}.
More precisely, find $\phi_{n1}, \ldots, \phi_n$ to minimize the MSE of the forecast:

$$E(x_{n+1} - \phi_{n1}x_n - \ldots - \phi_{nn}x_1)^2.$$

What are the Normal Equations?

Table 3.1. Behavior of the ACF and PACF for ARMA Models

	AR(p)	MA(q)	ARMA(p,q)
ACF	Tails off	Cuts off after lag q	Tails off
PACF	Cuts off after lag p	Tails off	Tails off

Example 3.17 Preliminary Analysis of the Recruitment Series

We consider the problem of modeling the Recruitment series shown in Figure 1.5. There are 453 months of observed recruitment ranging over the years 1950-1987. The ACF and the PACF given in Figure 3.5 are consistent with the behavior of an AR(2). The ACF has cycles corresponding roughly to a 12-month period, and the PACF has large values for $h = 1, 2$ and then is essentially zero for higher order lags. Based on Table 3.1, these results suggest that a second-order ($p = 2$) autoregressive model might provide a good fit. Although we will discuss estimation in detail in §3.6, we ran a regression (see §2.2) using the data triplets $\{(x; z_1, z_2) : (x_3; x_2, x_1), (x_4; x_3, x_2), \ldots, (x_{453}; x_{452}, x_{451})\}$ to fit a model of the form

$$x_t = \phi_0 + \phi_1 x_{t-1} + \phi_2 x_{t-2} + w_t$$

for $t = 3, 4, \ldots, 453$. The values of the estimates were $\widehat{\phi}_0 = 6.74_{(1.11)}$, $\widehat{\phi}_1 = 1.35_{(.04)}$, $\widehat{\phi}_2 = -.46_{(.04)}$, and $\widehat{\sigma}_w^2 = 89.72$, where the estimated standard errors are in parentheses.

The following R code can be used for this analysis. We use the script `acf2` to print and plot the ACF and PACF; see Appendix R for details.

```
acf2(rec, 48)      # will produce values and a graphic
(regr = ar.ols(rec, order=2, demean=FALSE, intercept=TRUE))
regr$asy.se.coef  # standard errors of the estimates
```

3.5 Forecasting

In forecasting, the goal is to predict future values of a time series, x_{n+m}, $m = 1, 2, \ldots$, based on the data collected to the present, $\boldsymbol{x} = \{x_n, x_{n-1}, \ldots, x_1\}$. Throughout this section, we will assume x_t is stationary and the model parameters are known. The problem of forecasting when the model parameters are unknown will be discussed in the next section; also, see Problem 3.26. The minimum mean square error predictor of x_{n+m} is

$$x_{n+m}^n = E(x_{n+m} \mid \boldsymbol{x}) \tag{3.57}$$

because the conditional expectation minimizes the mean square error

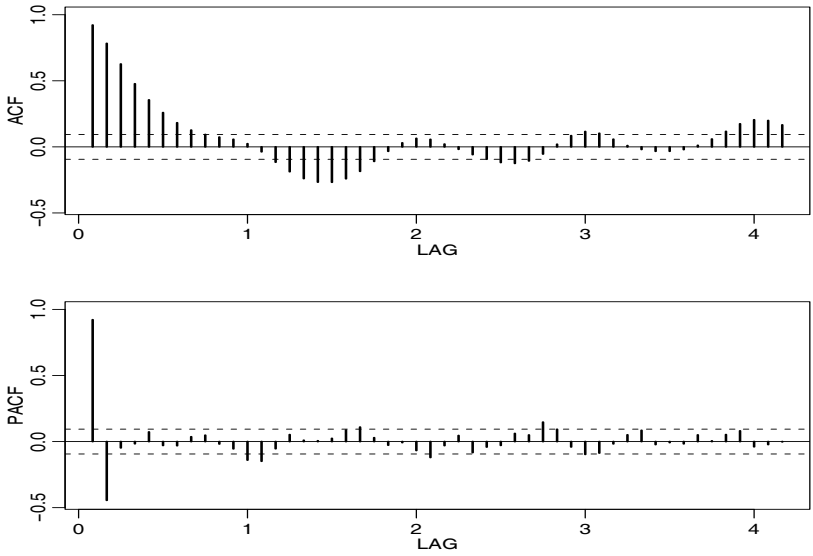


Fig. 3.5. ACF and PACF of the Recruitment series. Note that the lag axes are in terms of season (12 months in this case).

$$E\left[x_{n+m} - g(\boldsymbol{x})\right]^2, \tag{3.58}$$

where $g(\boldsymbol{x})$ is a function of the observations $\boldsymbol{x}$; see Problem 3.14.

First, we will restrict attention to predictors that are linear functions of the data, that is, predictors of the form

$$x^n_{n+m} = \alpha_0 + \sum_{k=1}^{n} \alpha_k x_k, \tag{3.59}$$

where $\alpha_0, \alpha_1, \ldots, \alpha_n$ are real numbers. Linear predictors of the form (3.59) that minimize the mean square prediction error (3.58) are called best linear predictors (BLPs). As we shall see, linear prediction depends only on the second-order moments of the process, which are easy to estimate from the data. Much of the material in this section is enhanced by the theoretical material presented in Appendix B. For example, Theorem B.3 states that if the process is Gaussian, minimum mean square error predictors and best linear predictors are the same. The following property, which is based on the Projection Theorem, Theorem B.1 of Appendix B, is a key result.

Property 3.3 Best Linear Prediction for Stationary Processes

Given data $x_1, \ldots, x_n$*, the best linear predictor,* $x^n_{n+m} = \alpha_0 + \sum_{k=1}^{n} \alpha_k x_k$*, of* x_{n+m}*, for* $m \geq 1$*, is found by solving*

$$E\left[\left(x_{n+m} - x^n_{n+m}\right) x_k\right] = 0, \quad k = 0, 1, \ldots, n, \tag{3.60}$$

where $x_0 = 1$*, for* $\alpha_0, \alpha_1, \ldots \alpha_n$*.*

The equations specified in (3.60) are called the prediction equations, and they are used to solve for the coefficients $\{\alpha_0, \alpha_1, \ldots, \alpha_n\}$. If $E(x_t) = \mu$, the first equation ($k = 0$) of (3.60) implies

$$E(x^n_{n+m}) = E(x_{n+m}) = \mu.$$

Thus, taking expectation in (3.59), we have

$$\mu = \alpha_0 + \sum_{k=1}^{n} \alpha_k \mu \qquad \text{or} \qquad \alpha_0 = \mu\Big(1 - \sum_{k=1}^{n} \alpha_k\Big).$$

Hence, the form of the BLP is

$$x^n_{n+m} = \mu + \sum_{k=1}^{n} \alpha_k (x_k - \mu).$$

Thus, until we discuss estimation, there is no loss of generality in considering the case that $\mu = 0$, in which case, $\alpha_0 = 0$.

First, consider one-step-ahead prediction. That is, given $\{x_1, \ldots, x_n\}$, we wish to forecast the value of the time series at the next time point, x_{n+1}. The BLP of x_{n+1} is of the form

$$x^n_{n+1} = \phi_{n1} x_n + \phi_{n2} x_{n-1} + \cdots + \phi_{nn} x_1, \tag{3.61}$$

where, for purposes that will become clear shortly, we have written α_k in (3.59), as $\phi_{n,n+1-k}$ in (3.61), for $k = 1, \ldots, n$. Using Property 3.3, the coefficients $\{\phi_{n1}, \phi_{n2}, \ldots, \phi_{nn}\}$ satisfy

$$E\Big[\Big(x_{n+1} - \sum_{j=1}^{n} \phi_{nj} x_{n+1-j}\Big) x_{n+1-k}\Big] = 0, \quad k = 1, \ldots, n,$$

or

$$\sum_{j=1}^{n} \phi_{nj} \gamma(k-j) = \gamma(k), \quad k = 1, \ldots, n. \tag{3.62}$$

The prediction equations (3.62) can be written in matrix notation as

$$\Gamma_n \boldsymbol{\phi}_n = \boldsymbol{\gamma}_n, \tag{3.63}$$

where $\Gamma_n = \{\gamma(k-j)\}_{j,k=1}^n$ is an $n \times n$ matrix, $\boldsymbol{\phi}_n = (\phi_{n1}, \ldots, \phi_{nn})'$ is an $n \times 1$ vector, and $\boldsymbol{\gamma}_n = (\gamma(1), \ldots, \gamma(n))'$ is an $n \times 1$ vector.

The matrix Γ_n is nonnegative definite. If Γ_n is singular, there are many solutions to (3.63), but, by the Projection Theorem (Theorem B.1), x^n_{n+1} is unique. If Γ_n is nonsingular, the elements of $\boldsymbol{\phi}_n$ are unique, and are given by

$$\boldsymbol{\phi}_n = \Gamma_n^{-1} \boldsymbol{\gamma}_n. \tag{3.64}$$

For ARMA models, the fact that $\sigma_w^2 > 0$ and $\gamma(h) \to 0$ as $h \to \infty$ is enough to ensure that Γ_n is positive definite (Problem 3.12). It is sometimes convenient to write the one-step-ahead forecast in vector notation

$$x_{n+1}^n = \boldsymbol{\phi}_n' \boldsymbol{x}, \tag{3.65}$$

where $\boldsymbol{x} = (x_n, x_{n-1}, \ldots, x_1)'$.

The mean square one-step-ahead prediction error is

$$P_{n+1}^n = E(x_{n+1} - x_{n+1}^n)^2 = \gamma(0) - \boldsymbol{\gamma}_n' \Gamma_n^{-1} \boldsymbol{\gamma}_n. \tag{3.66}$$

To verify (3.66) using (3.64) and (3.65),

$$\begin{aligned} E(x_{n+1} - x_{n+1}^n)^2 &= E(x_{n+1} - \boldsymbol{\phi}_n' \boldsymbol{x})^2 = E(x_{n+1} - \boldsymbol{\gamma}_n' \Gamma_n^{-1} \boldsymbol{x})^2 \\ &= E(x_{n+1}^2 - 2\boldsymbol{\gamma}_n' \Gamma_n^{-1} \boldsymbol{x} x_{n+1} + \boldsymbol{\gamma}_n' \Gamma_n^{-1} \boldsymbol{x}\boldsymbol{x}' \Gamma_n^{-1} \boldsymbol{\gamma}_n) \\ &= \gamma(0) - 2\boldsymbol{\gamma}_n' \Gamma_n^{-1} \boldsymbol{\gamma}_n + \boldsymbol{\gamma}_n' \Gamma_n^{-1} \Gamma_n \Gamma_n^{-1} \boldsymbol{\gamma}_n \\ &= \gamma(0) - \boldsymbol{\gamma}_n' \Gamma_n^{-1} \boldsymbol{\gamma}_n. \end{aligned}$$

Example 3.18 Prediction for an AR(2)

Suppose we have a causal AR(2) process $x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + w_t$, and one observation x_1. Then, using equation (3.64), the one-step-ahead prediction of x_2 based on x_1 is

$$x_2^1 = \phi_{11} x_1 = \frac{\gamma(1)}{\gamma(0)} x_1 = \rho(1) x_1.$$

Now, suppose we want the one-step-ahead prediction of x_3 based on two observations x_1 and x_2; i.e., $x_3^2 = \phi_{21} x_2 + \phi_{22} x_1$. We could use (3.62)

$$\begin{aligned} \phi_{21}\gamma(0) + \phi_{22}\gamma(1) &= \gamma(1) \\ \phi_{21}\gamma(1) + \phi_{22}\gamma(0) &= \gamma(2) \end{aligned}$$

to solve for ϕ_{21} and ϕ_{22}, or use the matrix form in (3.64) and solve

$$\begin{pmatrix} \phi_{21} \\ \phi_{22} \end{pmatrix} = \begin{pmatrix} \gamma(0) & \gamma(1) \\ \gamma(1) & \gamma(0) \end{pmatrix}^{-1} \begin{pmatrix} \gamma(1) \\ \gamma(2) \end{pmatrix},$$

but, it should be apparent from the model that $x_3^2 = \phi_1 x_2 + \phi_2 x_1$. Because $\phi_1 x_2 + \phi_2 x_1$ satisfies the prediction equations (3.60),

$$\begin{aligned} E\{[x_3 - (\phi_1 x_2 + \phi_2 x_1)]x_1\} &= E(w_3 x_1) = 0, \\ E\{[x_3 - (\phi_1 x_2 + \phi_2 x_1)]x_2\} &= E(w_3 x_2) = 0, \end{aligned}$$

it follows that, indeed, $x_3^2 = \phi_1 x_2 + \phi_2 x_1$, and by the uniqueness of the coefficients in this case, that $\phi_{21} = \phi_1$ and $\phi_{22} = \phi_2$. Continuing in this way, it is easy to verify that, for $n \geq 2$,

$$x_{n+1}^n = \phi_1 x_n + \phi_2 x_{n-1}.$$

That is, $\phi_{n1} = \phi_1, \phi_{n2} = \phi_2$, and $\phi_{nj} = 0$, for $j = 3, 4, \ldots, n$.

From Example 3.18, it should be clear (Problem 3.40) that, if the time series is a causal AR(p) process, then, for $n \geq p$,

$$x_{n+1}^n = \phi_1 x_n + \phi_2 x_{n-1} + \cdots + \phi_p x_{n-p+1}. \tag{3.67}$$

For ARMA models in general, the prediction equations will not be as simple as the pure AR case. In addition, for n large, the use of (3.64) is prohibitive because it requires the inversion of a large matrix. There are, however, iterative solutions that do not require any matrix inversion. In particular, we mention the recursive solution due to Levinson (1947) and Durbin (1960).

Property 3.4 The Durbin–Levinson Algorithm

Equations (3.64) and (3.66) can be solved iteratively as follows:

$$\phi_{00} = 0, \quad P_1^0 = \gamma(0). \tag{3.68}$$

For $n \geq 1$,

$$\phi_{nn} = \frac{\rho(n) - \sum_{k=1}^{n-1} \phi_{n-1,k}\ \rho(n-k)}{1 - \sum_{k=1}^{n-1} \phi_{n-1,k}\ \rho(k)}, \quad P_{n+1}^n = P_n^{n-1}(1 - \phi_{nn}^2), \tag{3.69}$$

where, for $n \geq 2$,

$$\phi_{nk} = \phi_{n-1,k} - \phi_{nn}\phi_{n-1,n-k}, \quad k = 1, 2, \ldots, n-1. \tag{3.70}$$

The proof of Property 3.4 is left as an exercise; see Problem 3.13.

Example 3.19 Using the Durbin–Levinson Algorithm

To use the algorithm, start with $\phi_{00} = 0$, $P_1^0 = \gamma(0)$. Then, for $n = 1$,

$$\phi_{11} = \rho(1), \quad P_2^1 = \gamma(0)[1 - \phi_{11}^2].$$

For $n = 2$,

$$\phi_{22} = \frac{\rho(2) - \phi_{11}\ \rho(1)}{1 - \phi_{11}\ \rho(1)}, \ \phi_{21} = \phi_{11} - \phi_{22}\phi_{11},$$
$$P_3^2 = P_2^1[1 - \phi_{22}^2] = \gamma(0)[1 - \phi_{11}^2][1 - \phi_{22}^2].$$

For $n = 3$,

$$\phi_{33} = \frac{\rho(3) - \phi_{21}\ \rho(2) - \phi_{22}\ \rho(1)}{1 - \phi_{21}\ \rho(1) - \phi_{22}\ \rho(2)},$$
$$\phi_{32} = \phi_{22} - \phi_{33}\phi_{21}, \ \phi_{31} = \phi_{21} - \phi_{33}\phi_{22},$$
$$P_4^3 = P_3^2[1 - \phi_{33}^2] = \gamma(0)[1 - \phi_{11}^2][1 - \phi_{22}^2][1 - \phi_{33}^2],$$

and so on. Note that, in general, the standard error of the one-step-ahead forecast is the square root of

$$P_{n+1}^n = \gamma(0) \prod_{j=1}^{n} [1 - \phi_{jj}^2]. \tag{3.71}$$

An important consequence of the Durbin–Levinson algorithm is (see Problem 3.13) as follows.

Property 3.5 Iterative Solution for the PACF

The PACF of a stationary process x_t, can be obtained iteratively via (3.69) as ϕ_{nn}, for $n = 1, 2, \ldots$.

Using Property 3.5 and putting $n = p$ in (3.61) and (3.67), it follows that for an AR(p) model,

$$\begin{aligned} x^p_{p+1} &= \phi_{p1}\, x_p + \phi_{p2}\, x_{p-1} + \cdots + \phi_{pp}\, x_1 \\ &= \phi_1\, x_p + \phi_2\, x_{p-1} + \cdots + \phi_p\, x_1. \end{aligned} \tag{3.72}$$

Result (3.72) shows that for an AR(p) model, the partial autocorrelation coefficient at lag p, ϕ_{pp}, is also the last coefficient in the model, ϕ_p, as was claimed in Example 3.15.

Example 3.20 The PACF of an AR(2)

We will use the results of Example 3.19 and Property 3.5 to calculate the first three values, ϕ_{11}, ϕ_{22}, ϕ_{33}, of the PACF. Recall from Example 3.9 that $\rho(h) - \phi_1\rho(h-1) - \phi_2\rho(h-2) = 0$ for $h \geq 1$. When $h = 1, 2, 3$, we have $\rho(1) = \phi_1/(1-\phi_2)$, $\rho(2) = \phi_1\rho(1) + \phi_2$, $\rho(3) - \phi_1\rho(2) - \phi_2\rho(1) = 0$. Thus,

$$\begin{aligned} \phi_{11} &= \rho(1) = \frac{\phi_1}{1-\phi_2} \\ \phi_{22} &= \frac{\rho(2) - \rho(1)^2}{1 - \rho(1)^2} = \frac{\left[\phi_1\left(\frac{\phi_1}{1-\phi_2}\right) + \phi_2\right] - \left(\frac{\phi_1}{1-\phi_2}\right)^2}{1 - \left(\frac{\phi_1}{1-\phi_2}\right)^2} = \phi_2 \\ \phi_{21} &= \rho(1)[1-\phi_2] = \phi_1 \\ \phi_{33} &= \frac{\rho(3) - \phi_1\rho(2) - \phi_2\rho(1)}{1 - \phi_1\rho(1) - \phi_2\rho(2)} = 0. \end{aligned}$$

Notice that, as shown in (3.72), $\phi_{22} = \phi_2$ for an AR(2) model.

So far, we have concentrated on one-step-ahead prediction, but Property 3.3 allows us to calculate the BLP of x_{n+m} for any $m \geq 1$. Given data, $\{x_1, \ldots, x_n\}$, the m-step-ahead predictor is

$$x^n_{n+m} = \phi^{(m)}_{n1} x_n + \phi^{(m)}_{n2} x_{n-1} + \cdots + \phi^{(m)}_{nn} x_1, \tag{3.73}$$

where $\{\phi^{(m)}_{n1}, \phi^{(m)}_{n2}, \ldots, \phi^{(m)}_{nn}\}$ satisfy the prediction equations,

$$\sum_{j=1}^{n} \phi^{(m)}_{nj} E(x_{n+1-j} x_{n+1-k}) = E(x_{n+m} x_{n+1-k}), \quad k = 1, \ldots, n,$$

or

Example 3.23 Forecasting an ARMA$(1,1)$ Series

Given data $x_1, \ldots, x_n$, for forecasting purposes, write the model as

$$x_{n+1} = \phi x_n + w_{n+1} + \theta w_n.$$

Then, based on (3.92), the one-step-ahead truncated forecast is

$$\widetilde{x}^n_{n+1} = \phi x_n + 0 + \theta \widetilde{w}^n_n.$$

For $m \geq 2$, we have

$$\widetilde{x}^n_{n+m} = \phi \widetilde{x}^n_{n+m-1},$$

which can be calculated recursively, $m = 2, 3, \ldots$.

To calculate $\widetilde{w}^n_n$, which is needed to initialize the successive forecasts, the model can be written as $w_t = x_t - \phi x_{t-1} - \theta w_{t-1}$ for $t = 1, \ldots, n$. For truncated forecasting using (3.92), put $\widetilde{w}^n_0 = 0$, $x_0 = 0$, and then iterate the errors forward in time

$$\widetilde{w}^n_t = x_t - \phi x_{t-1} - \theta \widetilde{w}^n_{t-1}, \quad t = 1, \ldots, n.$$

The approximate forecast variance is computed from (3.86) using the ψ-weights determined as in Example 3.11. In particular, the ψ-weights satisfy $\psi_j = (\phi + \theta)\phi^{j-1}$, for $j \geq 1$. This result gives

$$P^n_{n+m} = \sigma^2_w \left[1 + (\phi + \theta)^2 \sum_{j=1}^{m-1} \phi^{2(j-1)} \right] = \sigma^2_w \left[1 + \frac{(\phi + \theta)^2 (1 - \phi^{2(m-1)})}{(1 - \phi^2)} \right].$$

To assess the precision of the forecasts, prediction intervals are typically calculated along with the forecasts. In general, $(1 - \alpha)$ prediction intervals are of the form

$$x^n_{n+m} \pm c_{\frac{\alpha}{2}} \sqrt{P^n_{n+m}}, \tag{3.93}$$

where $c_{\alpha/2}$ is chosen to get the desired degree of confidence. For example, if the process is Gaussian, then choosing $c_{\alpha/2} = 2$ will yield an approximate 95% prediction interval for x_{n+m}. If we are interested in establishing prediction intervals over more than one time period, then $c_{\alpha/2}$ should be adjusted appropriately, for example, by using Bonferroni's inequality [see (4.55) in Chapter 4 or Johnson and Wichern, 1992, Chapter 5].

Example 3.24 Forecasting the Recruitment Series

Using the parameter estimates as the actual parameter values, Figure 3.6 shows the result of forecasting the Recruitment series given in Example 3.17 over a 24-month horizon, $m = 1, 2, \ldots, 24$. The actual forecasts are calculated as

$$x^n_{n+m} = 6.74 + 1.35 x^n_{n+m-1} - .46 x^n_{n+m-2}$$

for $n = 453$ and $m = 1, 2, \ldots, 12$. Recall that $x^s_t = x_t$ when $t \leq s$. The forecasts errors P^n_{n+m} are calculated using (3.86). Recall that $\widehat{\sigma}^2_w = 89.72$,

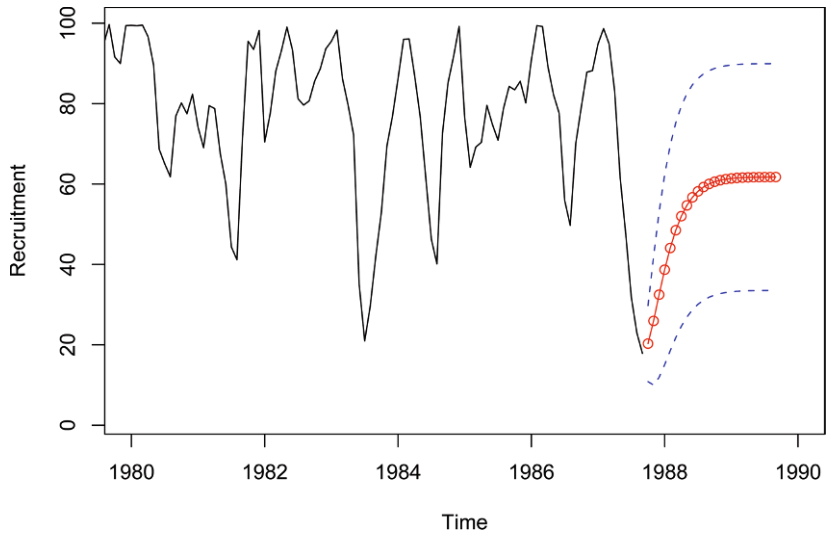


Fig. 3.6. Twenty-four month forecasts for the Recruitment series. The actual data shown are from about January 1980 to September 1987, and then the forecasts plus and minus one standard error are displayed.

and using (3.40) from Example 3.11, we have $\psi_j = 1.35\psi_{j-1} - .46\psi_{j-2}$ for $j \geq 2$, where $\psi_0 = 1$ and $\psi_1 = 1.35$. Thus, for $n = 453$,

$$\begin{aligned} P^n_{n+1} &= 89.72, \\ P^n_{n+2} &= 89.72(1 + 1.35^2), \\ P^n_{n+3} &= 89.72(1 + 1.35^2 + [1.35^2 - .46]^2), \end{aligned}$$

and so on.

Note how the forecast levels off quickly and the prediction intervals are wide, even though in this case the forecast limits are only based on one standard error; that is, $x^n_{n+m} \pm \sqrt{P^n_{n+m}}$.

To reproduce the analysis and Figure 3.6, use the following commands:

```
regr = ar.ols(rec, order=2, demean=FALSE, intercept=TRUE)
fore = predict(regr, n.ahead=24)
ts.plot(rec, fore$pred, col=1:2, xlim=c(1980,1990),
    ylab="Recruitment")
lines(fore$pred, type="p", col=2)
lines(fore$pred+fore$se, lty="dashed", col=4)
lines(fore$pred-fore$se, lty="dashed", col=4)
```

We complete this section with a brief discussion of backcasting. In backcasting, we want to predict x_{1-m}, for $m = 1, 2, \ldots$, based on the data $\{x_1, \ldots, x_n\}$. Write the backcast as

3.7 Integrated Models for Nonstationary Data

In Chapters 1 and 2, we saw that if x_t is a random walk, $x_t = x_{t-1} + w_t$, then by differencing x_t, we find that $\nabla x_t = w_t$ is stationary. In many situations, time series can be thought of as being composed of two components, a nonstationary trend component and a zero-mean stationary component. For example, in §2.2 we considered the model

$$x_t = \mu_t + y_t, \tag{3.142}$$

where $\mu_t = \beta_0 + \beta_1 t$ and y_t is stationary. Differencing such a process will lead to a stationary process:

$$\nabla x_t = x_t - x_{t-1} = \beta_1 + y_t - y_{t-1} = \beta_1 + \nabla y_t.$$

Another model that leads to first differencing is the case in which μ_t in (3.142) is stochastic and slowly varying according to a random walk. That is,

$$\mu_t = \mu_{t-1} + v_t$$

where v_t is stationary. In this case,

$$\nabla x_t = v_t + \nabla y_t,$$

is stationary. If μ_t in (3.142) is a k-th order polynomial, $\mu_t = \sum_{j=0}^{k} \beta_j t^j$, then (Problem 3.27) the differenced series $\nabla^k y_t$ is stationary. Stochastic trend models can also lead to higher order differencing. For example, suppose

$$\mu_t = \mu_{t-1} + v_t \quad \text{and} \quad v_t = v_{t-1} + e_t,$$

where e_t is stationary. Then, $\nabla x_t = v_t + \nabla y_t$ is not stationary, but

$$\nabla^2 x_t = e_t + \nabla^2 y_t$$

is stationary.

The integrated ARMA, or ARIMA, model is a broadening of the class of ARMA models to include differencing.

Definition 3.11 *A process x_t is said to be* **ARIMA(p, d, q)** *if*

$$\nabla^d x_t = (1 - B)^d x_t$$

is ARMA(p, q). In general, we will write the model as

$$\phi(B)(1 - B)^d x_t = \theta(B) w_t. \tag{3.143}$$

If $E(\nabla^d x_t) = \mu$, we write the model as

$$\phi(B)(1 - B)^d x_t = \delta + \theta(B) w_t,$$

where $\delta = \mu(1 - \phi_1 - \cdots - \phi_p)$.

Because of the nonstationarity, care must be taken when deriving forecasts. For the sake of completeness, we discuss this issue briefly here, but we stress the fact that both the theoretical and computational aspects of the problem are best handled via state-space models. We discuss the theoretical details in Chapter 6. For information on the state-space based computational aspects in R, see the ARIMA help files (`?arima` and `?predict.Arima`); our scripts `sarima` and `sarima.for` are basically front ends for these R scripts.

It should be clear that, since $y_t = \nabla^d x_t$ is ARMA, we can use §3.5 methods to obtain forecasts of y_t, which in turn lead to forecasts for x_t. For example, if $d = 1$, given forecasts y^n_{n+m} for $m = 1, 2, \ldots$, we have $y^n_{n+m} = x^n_{n+m} - x^n_{n+m-1}$, so that

$$x^n_{n+m} = y^n_{n+m} + x^n_{n+m-1}$$

with initial condition $x^n_{n+1} = y^n_{n+1} + x_n$ (noting $x^n_n = x_n$).

It is a little more difficult to obtain the prediction errors P^n_{n+m}, but for large n, the approximation used in §3.5, equation (3.86), works well. That is, the mean-squared prediction error can be approximated by

$$P^n_{n+m} = \sigma_w^2 \sum_{j=0}^{m-1} \psi_j^{*2}, \tag{3.144}$$

where ψ^*_j is the coefficient of z^j in $\psi^*(z) = \theta(z)/\phi(z)(1-z)^d$.

To better understand integrated models, we examine the properties of some simple cases; Problem 3.29 covers the ARIMA$(1, 1, 0)$ case.

Example 3.36 Random Walk with Drift

To fix ideas, we begin by considering the random walk with drift model first presented in Example 1.11, that is,

$$x_t = \delta + x_{t-1} + w_t,$$

for $t = 1, 2, \ldots$, and $x_0 = 0$. Technically, the model is not ARIMA, but we could include it trivially as an ARIMA$(0, 1, 0)$ model. Given data $x_1, \ldots, x_n$, the one-step-ahead forecast is given by

$$x^n_{n+1} = E(x_{n+1} \mid x_n, \ldots, x_1) = E(\delta + x_n + w_{n+1} \mid x_n, \ldots, x_1) = \delta + x_n.$$

The two-step-ahead forecast is given by $x^n_{n+2} = \delta + x^n_{n+1} = 2\delta + x_n$, and consequently, the m-step-ahead forecast, for $m = 1, 2, \ldots$, is

$$x^n_{n+m} = m\,\delta + x_n, \tag{3.145}$$

To obtain the forecast errors, it is convenient to recall equation (1.4), i.e., $x_n = n\,\delta + \sum_{j=1}^n w_j$, in which case we may write

$$x_{n+m} = (n+m)\,\delta + \sum_{j=1}^{n+m} w_j = m\,\delta + x_n + \sum_{j=n+1}^{n+m} w_j.$$

From this it follows that the m-step-ahead prediction error is given by

$$P_{n+m}^{n} = E(x_{n+m} - x_{n+m}^{n})^2 = E\Big(\sum_{j=n+1}^{n+m} w_j\Big)^2 = m\,\sigma_w^2. \tag{3.146}$$

Hence, unlike the stationary case (see Example 3.22), as the forecast horizon grows, the prediction errors, (3.146), increase without bound and the forecasts follow a straight line with slope δ emanating from x_n. We note that (3.144) is exact in this case because $\psi^*(z) = 1/(1-z) = \sum_{j=0}^{\infty} z^j$ for $|z| < 1$, so that $\psi_j^* = 1$ for all j.

The w_t are Gaussian, so estimation is straightforward because the differenced data, say $y_t = \nabla x_t$, are independent and identically distributed normal variates with mean δ and variance σ_w^2. Consequently, optimal estimates of δ and σ_w^2 are the sample mean and variance of the y_t, respectively.

Example 3.37 IMA$(1,1)$ and EWMA

The ARIMA(0,1,1), or IMA(1,1) model is of interest because many economic time series can be successfully modeled this way. In addition, the model leads to a frequently used, and abused, forecasting method called exponentially weighted moving averages (EWMA). We will write the model as

$$x_t = x_{t-1} + w_t - \lambda w_{t-1}, \tag{3.147}$$

with $|\lambda| < 1$, for $t = 1, 2, \ldots$, and $x_0 = 0$, because this model formulation is easier to work with here, and it leads to the standard representation for EWMA. We could have included a drift term in (3.147), as was done in the previous example, but for the sake of simplicity, we leave it out of the discussion. If we write

$$y_t = w_t - \lambda w_{t-1},$$

we may write (3.147) as $x_t = x_{t-1} + y_t$. Because $|\lambda| < 1$, y_t has an invertible representation, $y_t = \sum_{j=1}^{\infty} \lambda^j y_{t-j} + w_t$, and substituting $y_t = x_t - x_{t-1}$, we may write

$$x_t = \sum_{j=1}^{\infty} (1-\lambda)\lambda^{j-1} x_{t-j} + w_t. \tag{3.148}$$

as an approximation for large t (put $x_t = 0$ for $t \le 0$). Verification of (3.148) is left to the reader (Problem 3.28). Using the approximation (3.148), we have that the approximate one-step-ahead predictor, using the notation of §3.5, is

$$\begin{aligned} \tilde{x}_{n+1} &= \sum_{j=1}^{\infty} (1-\lambda)\lambda^{j-1} x_{n+1-j} \\ &= (1-\lambda)x_n + \lambda \sum_{j=1}^{\infty} (1-\lambda)\lambda^{j-1} x_{n-j} \\ &= (1-\lambda)x_n + \lambda \tilde{x}_n. \end{aligned} \tag{3.149}$$

From (3.149), we see that the new forecast is a linear combination of the old forecast and the new observation. Based on (3.149) and the fact that we only observe $x_1, \ldots, x_n$, and consequently $y_1, \ldots, y_n$ (because $y_t = x_t - x_{t-1}$; $x_0 = 0$), the truncated forecasts are

$$\tilde{x}^n_{n+1} = (1-\lambda)x_n + \lambda\tilde{x}^{n-1}_n, \quad n \geq 1, \tag{3.150}$$

with $\tilde{x}^0_1 = x_1$ as an initial value. The mean-square prediction error can be approximated using (3.144) by noting that $\psi^*(z) = (1-\lambda z)/(1-z) = 1 + (1-\lambda)\sum_{j=1}^{\infty} z^j$ for $|z| < 1$; consequently, for large n, (3.144) leads to

$$P^n_{n+m} \approx \sigma^2_w[1 + (m-1)(1-\lambda)^2].$$

In EWMA, the parameter $1-\lambda$ is often called the smoothing parameter and is restricted to be between zero and one. Larger values of λ lead to smoother forecasts. This method of forecasting is popular because it is easy to use; we need only retain the previous forecast value and the current observation to forecast the next time period. Unfortunately, as previously suggested, the method is often abused because some forecasters do not verify that the observations follow an IMA$(1,1)$ process, and often arbitrarily pick values of λ. In the following, we show how to generate 100 observations from an IMA(1,1) model with $\lambda = -\theta = .8$ and then calculate and display the fitted EWMA superimposed on the data. This is accomplished using the Holt-Winters command in R (see the help file `?HoltWinters` for details; no output is shown):

```
set.seed(666)
x = arima.sim(list(order = c(0,1,1), ma = -0.8), n = 100)
(x.ima = HoltWinters(x, beta=FALSE, gamma=FALSE))  # α below is 1 − λ
  Smoothing parameter: alpha: 0.1663072
plot(x.ima)
```

3.8 Building ARIMA Models

There are a few basic steps to fitting ARIMA models to time series data. These steps involve plotting the data, possibly transforming the data, identifying the dependence orders of the model, parameter estimation, diagnostics, and model choice. First, as with any data analysis, we should construct a time plot of the data, and inspect the graph for any anomalies. If, for example, the variability in the data grows with time, it will be necessary to transform the data to stabilize the variance. In such cases, the Box–Cox class of power transformations, equation (2.37), could be employed. Also, the particular application might suggest an appropriate transformation. For example, suppose a process evolves as a fairly small and stable percent-change, such as an investment. For example, we might have

$$x_t = (1 + p_t)x_{t-1},$$

Not part of the crowd

"Wisdom of crowds" is a 100-year-old idea that underpins many of today's online services. But while the concept is widely understood, its statistical fundamentals are not. **Carlos A. Gómez Grajales** explains

In an article published in 1907, the statistician Sir Francis Galton proposed that the accumulated knowledge of a crowd could be more accurate than individual predictions – even those of supposed experts.[1] To demonstrate this, he used data from a raffle held at the annual West of England Fat Stock and Poultry Exhibition. Some 800 tickets were purchased by competitors who hoped to win a prize by guessing the correct butchered weight of a selected ox.

Galton discovered that the average guess of all the entrants was remarkably close to the actual weight, and a closer match than the one made by the individual who won the competition. It was out by only 1 lb. – a remarkable feat considering that the ox weighed 1198 lb. At that moment, the term "wisdom of the crowds" was born.

More than a hundred years later, Galton's idea has become a cornerstone of the modern interconnected world. Thanks to the aggregated opinions of millions, I can choose the best movie to watch using Metacritic, plan where to get dinner after the movie based on Yelp reviews, and use crowd-sourced traffic data from Waze to find the quickest route home.

But the wisdom of crowds is not infallible. Consider SketchFactor, a crowd-sourced safety-mapping app that launched last August in the US. SketchFactor encourages users to share details of "sketchy" activities witnessed in a given location – though the definition of "sketchy" is fairly loose. SketchFactor would then use these eye-witness reports to help other users avoid potentially "dangerous" areas, and to plot "safe" routes to and from specific locations.

Unfortunately, the data is not quite accurate enough. A comparison of the available SketchFactor reports for Washington, DC and actual crime data from the Metropolitan Police Department revealed few overlaps.[2] Media reports then started labelling the app "racist". One reporter summarised it as "yet another app for avoiding non-white areas of your town" (tek.io/1w1Jymk).

SketchFactor has many problems – not least the inherent vagueness of what it seeks to measure. In keeping with many services that rely on user-generated data, it borrows the wisdom-of-the-crowds concept from Galton but forgets to apply fundamental statistics.

As a statistician, Galton would tell you that crowd-sourcing is just another way of gathering data. Therefore, in order to properly summarise the collected information, statistical analysis is required. To support current and future app developers, and to help them avoid the mistakes of SketchFactor and others of that ilk, I have put together this brief guide to some of the common statistical concepts that crowd-sourcing companies will need to bear in mind as they look to improve their services.

Image: luis_escriv/iStock/Thinkstock

Outliers

Galton's original exercise was useful because weight guesses followed a simple pattern, with most of the data points sticking reasonably close to the actual weight of the ox. Nobody guessed the animal would weigh 8 million pounds, nor did anyone hazard a guess of 11 ounces. Still, Galton excluded 13 guesses from his study – estimates that were, in his view, either defective or illegible. In this way, he eliminated the effect of possible outliers.

Do crowd-sourcing apps control for this? It would seem unlikely, even though outliers are and will be present in the data. In the case of SketchFactor, there could be any number of temporary situations that increase the perception of crime in an area but which are, in context, isolated incidents.

Many services that rely on user-generated data borrow the wisdom-of-the-crowds concept from Galton but forget to apply fundamental statistics

Then there are the jokers, such as the SketchFactor user who described the area around the Capitol in Washington, DC as "the most dangerous place in the world … Politicians everywhere! Hold on to your wallets!" Another "report" described a murder that took place in the Netflix series *House of Cards*. A third described a restaurant's pricing as "dangerous".

Without screening or monitoring, bad data will sneak in, no matter what. Crowd-sourcing companies need to invest some effort to fight this if they want to keep their information useful. Yelp has an algorithm that aims to reduce the effect of false positive reviews (bit.ly/1rrD5Yf), for example.

This is a serious issue, one that greatly impacts the results generated by crowd-sourcing services. But it is not a new problem. In his original article, Galton pointed out how the 6d raffle entry fee "deterred practical joking, and the hope of a prize and the joy of competition prompted each competitor to do his best".

There are no hard and fast rules for dealing with outliers. Some statisticians believe that excluding outliers, as Galton did, may not be the best course of action. Others recommend studying the effect of outliers before making a call on what to do with them. In the case of crowd-sourcing apps and services, weighting observations for reliability may be useful, thus improving the accuracy of the results by decreasing the impact of unusual situations or spoof reports and comments.

Dependence of observations

In Galton's ox data, observations were independent. Each competitor wrote his best guess down without any reference to what the last competitor had estimated. "The judgments were unbiased by passion and uninfluenced by oratory and the like", wrote Galton. Sadly for many crowd-sourcing apps, their users' opinions are not independent.

Just ask James Surowiecki, author of *The Wisdom of Crowds: Why the Many Are Smarter than the Few and How Collective Wisdom Shapes Business, Economies, Societies and Nations*. In his book, Surowiecki clearly exposes independence as one of the key issues that separate wise crowds form irrational ones.[3] He described the problem of non-independent crowds by using the term "imitation". This means that when people's opinions are influenced by others, their judgements become less useful, making the aggregate result less accurate each time.

Consider the SketchFactor user walking home one night, who decides to check the "sketchiness" of the area he is in. The aggregate view is that this is a bad neighbourhood, at which point it is likely that our user is going to become extremely alert and suspicious. A couple talking in a car might be misinterpreted as a drug deal in progress.

The end result is that a neighbourhood with a bad reputation may be more likely to receive bad reports over time, even when such a reputation is unwarranted and such activities are not taking place.

Most statistical analyses rely on independent observations, as it is one of the fundamentals associated with random samples. If your data shows signs of aggregation or correlation among certain observations, your model should account for this. For instance, time series models help us deal with non-independent data, in cases where the latest observations are correlated with other recent ones. Clustered observations (reports from the same state, perhaps) may also exhibit high levels of correlation that should be considered in the model.

Insufficient sample size

There is no such thing as a correct sample size – it really depends on what it is you are trying to measure. Galton aggregated close to 800

guesses to produce an average for the weight of an ox, but while many crowd-sourcing apps can count total users in the millions, their samples are not evenly spread. For instance, in an area of southeast Washington, DC, SketchFactor had, for a time, just a solitary review, one that allowed the place to be catalogued as "dangerously" sketchy as the user in question had reported being "followed and attacked by two males".

Sites like Rotten Tomatoes or Metacritic report the number of reviews that their aggregate scores are based on, to help users judge reliability. But that kind of information can be missing in other applications, and small sample sizes can seriously skew results.

Most crowd-sourcing services do not realise it, but representativeness is their greatest problem. Their users reflect only the opinions of a small proportion of the population

Statisticians know this, of course. Power and sample size calculations are done before any experiment starts, and sample sizes for surveys are usually planned in advance to achieve a desired level of precision.

In some instances, small sample sizes are unavoidable and there are approaches – such as non-parametric statistics and Bayesian analysis – that may produce useful contrasts from small samples. The secret to success is to realise that you cannot treat 10 observations in the same way as you would 5000. Thus, crowd-sourcing efforts may benefit from that kind of knowledge – either by using different algorithms where necessary or simply by acknowledging that they do not have enough information to present an adequate result.

Non-representative samples

Galton's raffle had a diversity of people participating in it; people with different backgrounds and knowledge. He wrote: "The competitors included butchers and farmers, some of whom were highly expert in judging the weight of cattle; others were probably guided by such information as they might pick up, and by their own fancies." We cannot say whether the competitors constituted a representative sample of the local population. However, we do know that the sample was diverse enough to make it effective.

Most crowd-sourcing services do not realise it, but representativeness is their greatest problem. Their users are not random samples (nor are their observations necessarily independent, as explained previously). As such, they usually reflect only the opinions of a small proportion of the population.

SketchFactor was available only on iPhones and iPads for its first couple of months. While users of these devices tend to have shared characteristics, tastes and desires, these traits are not a guaranteed match to those of the broader US population. This is a major issue for an app designed to measure perceptions of what is and is not "sketchy". It limits your data to what high-income, mostly white working people think.

Non-random samples are common in statistical studies, particularly in health research, where randomisation may not be totally possible. In these instances, analysts have to correct for selection bias using various methods. Still, the best option is to prevent selection bias as much as possible, thus ensuring some sort of random sample is available to improve representativeness.

Over 100 years ago, Galton established crowd-sourcing as a concept – the idea that together we can produce better and more accurate results. Networked technology has allowed that idea to flourish, but it is important to remember the statistical underpinnings of Galton's work. It is not enough simply to gather data; we need to analyse it in intelligent ways so as to extract useful information from millions of individual opinions. Galton showed us that the crowd can be wise, but if crowd-sourcing services wish to access that wisdom, they need to get smarter about the way they collect, analyse and report data.

References

1. Galton, F. (1907) Vox populi. *Nature*, **75**(1949), 450–451. http://galton.org/essays/1900-1911/galton-1907-vox-populi.pdf
2. Dewey, C. (2014) The many problems with SketchFactor, the new crime crowd-sourcing app that some are calling racist. *Washington Post*, 12 August. wapo.st/1utfWLY
3. Surowiecki, J. (2005) *The Wisdom of Crowds: Why the Many Are Smarter than the Few and How Collective Wisdom Shapes Business, Economies, Societies and Nations*. New York: Anchor.

Carlos A. Gómez Grajales is a Mexican certified statistician and consultant. He works for a consulting firm specialising in political studies, survey design and analytical tools for businesses and governments. His fields of expertise include complex survey analysis and statistical modelling