Fourier-autoregressive (F-AR) coefficient non-linear time-series model for forecasting asymmetric cyclical data

HIMADRI GHOSH¹, K A SARKAR² and PRAJNESHU³

Indian Agricultural Statistics Research Institute, New Delhi 110 012 India

Received: 20 May 2013; Accepted: 21 March 2014

Key words: Bootstrap, F-AR Model, Monte Carlo simulation, Oil sardine landings data, Test

The autoregressive (AR) modelling is generally employed for forecasting time-series data and finds excellent applications in describing those situations in which the present value of the variable depends linearly on its past values (Box et al. 2009). However, one drawback of this model is that the coefficients are assumed to be constants. Ludlow and Enders (2000) proposed a generalization of AR model of order 1 by allowing autoregressive coefficient to be a time-dependent deterministic function \( \alpha_t \). Although its form is not completely specified, behaviour of \( \alpha_t \) can generally be represented by a sufficiently long Fourier series (Bloomfield 2000). For example, if \( \alpha_t \) is an absolutely integrable function, for any desired level of accuracy, it is possible to write the Fourier-autoregressive (F-AR) model as

\[
y_t = a_t y_{t-1} + \varepsilon_t, \quad a_t = A_0 + \sum_{k=1}^{T/4} A_k \sin \frac{2 \pi k t}{T} + B_k \cos \frac{2 \pi k t}{T},
\]

where \( T/4 \) is the maximum number of frequencies contained in the process generating \( \alpha_t \) for modelling quarterly time-series data and, \( A_k, B_k, k=1,2,\ldots, T/4 \) are the Fourier coefficients. Unlike linear AR model, F-AR model is a non-linear time-series model.

Fun and Yao 2003, Milas et al. 2006) and its heartening aspect is that it is capable of describing asymmetric data (Amendola and Storti 2002, Kiani 2009), wherein average number of observations on the up cycle is different from the down cycle. This model can be applied for modelling and forecasting any asymmetric time-series data in Fishery and Animal Sciences pertaining to Export and import quantities/ amount data of animal products, or Milk, meat and wool production data, etc.

Fitting and forecasting of F-AR model: Method of least squares is used to estimate the parameters of F-AR model. Further, \( H_{0k} : A_k = 0, B_k = 0 \) against \( H_{1k} \) : at least one of \( A_k \) or \( B_k \) is nonzero, \( k=1, 2, \ldots, T/4 \) are tested using t-test with (T-2) degrees of freedom, where single pair of sine-cosine terms is taken in eq. (1). For identification of significant Fourier coefficients, critical values are calculated by performing a Monte Carlo simulation study. For sample size \( T \), each realization of sequence \( \{ y_t \} \) is assumed to follow independent and identical standard normal distribution. The simulations are repeated a large number of times, say 20000. From each of the simulated samples, t-statistics are calculated and a histogram is drawn by taking t-statistic along X-axis and frequencies along Y-axis. The critical value at 5% level is calculated such that 5% observations lie above that point. If value of t-statistic is greater than this critical value, test is significant at 5% level and the significant variables are then included in the model. Further, goodness-of-fit of the fitted model is examined using Akaikie Information Criterion (AIC) and Bayesian Information Criterion (BIC).

To obtain size of the test, Double bootstrap technique (Efron and Tibshirani 1993) is employed. To this end, outset t-statistic values are first obtained by parametric bootstrap method using estimated coefficients of F-AR model. Then, in the second-stage, simulated statistic values are computed based on samples drawn from null distribution. For each of the first-stage outset values, t-statistics are calculated and a histogram is drawn by taking t-statistic along X-axis and frequencies along Y-axis. The critical value at 5% level is calculated such that 5% observations lie above that point. If value of t-statistic is greater than this critical value, test is significant at 5% level and the significant variables are then included in the model. Further, goodness-of-fit of the fitted model is examined using Akaikie Information Criterion (AIC) and Bayesian Information Criterion (BIC).

For calculating power of the test, the first-stage outset t-statistic values are obtained along similar lines as above. The first-stage samples are considered as populations under alternate hypotheses. In the second-stage, regression is performed on each first-stage sample. Then, simulated statistic values are computed based on samples drawn again using parametric bootstrap approach. The same procedure as mentioned in last paragraph is repeated with ‘greater’
replaced by ‘less’ and ‘true’ by ‘not true’.

Relevant computer programs for performing all the above tasks were developed in SAS/IML, Ver. 9.2 software package and the same are appended as an Annexure. To compare the performance of F-AR model vis-à-vis AR model, one-step ahead forecasts for hold-out data are obtained from fitted models and Mean square forecast errors (MSFE) are computed.

**Statistical analysis:** Indian oil sardine (*Sardinella longiceps*) is one of the most important commercial fishes in India and can be found in the northern regions of the Indian Ocean. Twentysix years (1985–2010) quarterly oil sardine fish landings data (in tonnes) recorded at Central Marine Fisheries Research Institute, Kochi, India were used in the study. A perusal showed cyclical behaviour of the data and suggested asymmetry in the dataset.

In order to fit eq. (1), significant Fourier coefficients were identified by generating 20,000 data sets through Monte Carlo simulation and t-statistics were calculated. Further, histograms are exhibited in Fig. 1. The critical value at 5% level was calculated as 2.00. So, significant Fourier coefficients at 5% level were found as A1, B1, A2, B3, B6, A15 and B16 and the fitted F-AR model was obtained as

\[ y_t = -0.08 + 0.13 y_{t-1} + \left[ -0.87 \sin \left( \frac{2 \pi 1}{96} \right) \right] - 1.03 \cos \left( \frac{2 \pi 1}{96} \right) - 1.79 \sin \left( \frac{2 \pi 2}{96} \right) + 1.95 \cos \left( \frac{2 \pi 3}{96} \right) - 1.61 \cos \left( \frac{2 \pi 6}{96} \right) + 1.19 \sin \left( \frac{2 \pi 15}{96} \right) + 0.68 \cos \left( \frac{2 \pi 16}{96} \right) y_{t-1} + \epsilon_t, \]

and the AIC and BIC values were respectively computed as 21.07 and 25.24. Corresponding values for fitted AR model were computed as 2157.359 and 2159.913. Lower values of both the statistics reflected superiority of F-AR model over AR model for describing the data under consideration. To get a visual idea, the fitted F-AR model along with data is exhibited in Fig. 2. Subsequently, sizes of the tests were computed and the results for significant ones are reported in Table 1. By using double bootstrap methodology, corresponding power of the tests were also calculated and the results are reported in Table 1.

The quarter-wise forecasts for hold-out data for two years were also computed and the same are reported in Table 2. Finally, the MSFE value for F-AR model was computed as 41099124.32, which being less than that for AR model, viz. 995694711.90, showed that the F-AR model was superior to AR model for forecasting purposes also.

### Table 1. Computation of sizes (at 5% level) and powers of the tests

<table>
<thead>
<tr>
<th>Fourier coefficients</th>
<th>Test statistic</th>
<th>Size of the test</th>
<th>Power of the test</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>( t_{A1} )</td>
<td>0.395</td>
<td>0.75</td>
</tr>
<tr>
<td>B1</td>
<td>( t_{B1} )</td>
<td>0.336</td>
<td>0.86</td>
</tr>
<tr>
<td>A2</td>
<td>( t_{A2} )</td>
<td>0.413</td>
<td>0.77</td>
</tr>
<tr>
<td>B3</td>
<td>( t_{B3} )</td>
<td>0.392</td>
<td>0.81</td>
</tr>
<tr>
<td>B6</td>
<td>( t_{B6} )</td>
<td>0.395</td>
<td>0.86</td>
</tr>
<tr>
<td>A15</td>
<td>( t_{A15} )</td>
<td>0.338</td>
<td>0.79</td>
</tr>
<tr>
<td>B16</td>
<td>( t_{B16} )</td>
<td>0.392</td>
<td>0.91</td>
</tr>
</tbody>
</table>

### Table 2. Computation of forecasts using F-AR and AR models

<table>
<thead>
<tr>
<th>Quarter number/Year</th>
<th>Actual value (in tonnes)</th>
<th>Forecast by F-AR model</th>
<th>Forecast by AR model</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st / 2009</td>
<td>38635</td>
<td>44065.039</td>
<td>61321.740</td>
</tr>
<tr>
<td>2nd / 2009</td>
<td>14855</td>
<td>16168.642</td>
<td>61263.639</td>
</tr>
<tr>
<td>3rd / 2009</td>
<td>74225</td>
<td>71846.180</td>
<td>61272.648</td>
</tr>
<tr>
<td>4th / 2009</td>
<td>39712</td>
<td>43690.751</td>
<td>61271.251</td>
</tr>
<tr>
<td>1st / 2010</td>
<td>67470</td>
<td>66030.316</td>
<td>61271.647</td>
</tr>
<tr>
<td>2nd / 2010</td>
<td>31303</td>
<td>32517.648</td>
<td>61271.434</td>
</tr>
<tr>
<td>3rd / 2010</td>
<td>41523</td>
<td>30713.824</td>
<td>61271.439</td>
</tr>
<tr>
<td>4th / 2010</td>
<td>119047</td>
<td>114673.314</td>
<td>61271.438</td>
</tr>
</tbody>
</table>
SUMMARY

The aim of this study was to apply Fourier autoregressive (F-AR) model to describe and forecast asymmetric cyclical data. For carrying out statistical analysis, computer programs were developed using SAS, Ver. 9.2 software package. Twenty-six years (1985–2010) quarterly oil sardine fish landings data (in tonnes) recorded at Central Marine Fisheries Research Institute, Kochi, India were used. Superiority of F-AR model over AR model was demonstrated by developing one-step ahead forecasts for two years’ hold-out data. Its potential use is to develop optimal import and export policies for Oil sardines. This type of information would also go a long way in enabling the Fishing industry in optimization of its resources. Efficient Oil sardine management strategies need to be evolved in order to allocate optimum number of boats and trawlers, etc.

ACKNOWLEDGEMENT

Authors are grateful to the referee for valuable comments.

REFERENCES


ANNEXURE

Code for calculation of Fourier coefficients, Test statistics, p-Values, Size and Power of test

(i) Estimation of Fourier coefficients and calculation of t-statistics:
Proc import datafile = “PATH OF DATA FILE.EXTN” out = oilsardine dbms = EXTN replace;
sheet = ’output’; getnames = yes;
run;
%macro estimation (dataname =, i =);
proc iml;
use &dataname. ;
read all into y;
close &dataname.;
x = j(nrow(y)-1,2*(nrow(y)/&period.),0);
y_vec = y[2: nrow(y),1];
do t = 2 to nrow(y);
do k = 1 to 2*(nrow(y)/&period.);
if mod(k,2) ^= 0 then
   x[t-1,k] = sin(2*&pi;*t*k/nrow(y))*y[t-1];
else
   x[t-1,k] = cos(2*&pi;*t*k/nrow(y))*y[t-1];
end;
beta_est = inv(x´*x)*x´*y_vec;
residual = y_vec - x*beta_est;
ess = residual´*residual;
p = diaginv(inv(x´*x));
diag_sum_p = p[%+,:]; mse = ess/nrow(x)-ncol(x);
numerator_t = beta_est`; denominator_t = sqrt (mse*diag_sum_p);
t_statistics = numerator_t/denominator_t;
create t_value var {t_statistics}; append;
create test_value&i. var {t_statistics}; append;
quit;
proc transpose data = test_value&i. out = statistics&i. ;
var t_statistics;
run;
%mend estimation;

(ii) Monte-Carlo simulation:
%macro simulation (i =);
data _null_; x = round (ranuni (0)*10000); call symputx (‘seed’,x);
run;
data simulation&i.;
do time = 1 to &n. ;
y = rannor (&seed.); output;
end;
keep y; run;
%mend simulation;
%macro montecarlo_estimation;
%do i = 0%to &no_simulation.;
%if &i. = 0%then%do;
estimation (dataname = oilsardine,i=&i.);
%end;
%else%do;
simulation (i=&i.); estimation (dataname = simulation&i. ,i=&i.);
%end;
%append (base_data=statistics0,append_data=&syslast);
%end; data mc_statistics (drop = _name_);
set statistics0;
%end;
%mend montecarlo_estimation;

(iii) Double bootstrap:
%macro single_bootstrap (n_boot=, b=);
data _null_; x = round (ranuni(0)*10000); call symputx (‘seed’,x);
run;
data bootdata0;
set oilsardine; run;
data boot_data;
do sample = 1 to &n_boot.;
%end;
%mend single_bootstrap;
do i = 1 to nobs;
   x = round (ranuni(&seed.) * nobs);
   set bootdata&b. nobs = nobs point = x;
output;
end;
end;
stop;
run;
%mend single_bootstrap;
%macro double_bootstrap;
%do j = 1%to &first_bootsample.;
   %single_bootstrap (n_boot = &first_bootsample., b = 0);
   data bootdata&j.;
      set bootdata;
      if sample = &j.;
      keep y;
   run;
%end;
%do j = 1%to &first_bootsample.;
   %single_bootstrap (n_boot = &second_bootsample., b = &j.);
   %do b = 1%to &second_bootsample.;
      data bootdata&j.&b.;
         set bootdata;
         if sample = &b.;
         keep y;
      run;
   %end;
%end;
%mend double_bootstrap;
%macro bootstrap_estimation;
%do i = 0%to &first_bootsample.;
   %if &i. = 0%then%do;
      %estimation (dataname = oilsardine,i=&i.);
      %append (base_data=statistics0,append_data=&syslast);
   %end;
   %else%do;
      %estimation (dataname = bootdata&i.,i=&i.);
      %append (base_data=statistics0,append_data=&syslast);
      %do j = 1%to &second_bootsample.;
         %estimation (dataname = bootdata&i.&j.,i=&i.&j.);
         %append (base_data=statistics0,append_data=&syslast);
      %end;
   %end;
%end;
%mend bootstrap_estimation;
%macro macrobootstrap_estimation;
%do i = 1%to &no_coeff.;
   data indiv_coeff_p_interim; set final_mc_pdataset;
      keep col&i.;
   run;
   data size_of_test;
      set indiv_coeff_p_interim;
      if col&i. < 0.05 then five_percent = 1; else five_percent = 0;
      proc means data = size_of_test sum noprint; var five_percent;
      output out = sot&i. sum =/autoname ;
   run;
%mend macrobootstrap_estimation;
%do i = 1%to &second_bootsample.;
   %macro first_level_bootsample;
      %let j = 0;
      data first_boot_pdata1; set boot_final_data; if id = “fout”;
      run;
      %do i = 1%to &first_bootsample.;
         %do j = 0%to &second_bootsample.;
            data boot_p_value&i.&j.;
               set boot_final_data;
               if id = ”&i.&j.”;
            run;
            %append (base_data=first_boot_pdata1,append_data=&syslast);
      %end;
   %end;
%mend first_level_bootsample;
%macro second_level_bootsample;
%let k = 0;
%do i = 1%to &first_bootsample.;
   %do j = 0%to &second_bootsample.;
      data boot_p_value&i.&j.;
         set boot_final_data;
         if id = ”&i.&j.”;
      run;
      %append (base_data=boot_p_value&i.&k., append_data=&syslast);
      data second_boot_pdata1&i.&k.;
         set boot_p_value&i.&k.;
         if _n_ = 1 then delete; run;
      data s_boot_pdata&i.&k.;
         set second_boot_pdata1&i.&k.;
         drop id;
      run;%end;
%end;
%mend second_level_bootsample;
%do i = 1%to &no_coeff.;
   data statistics; set &dataname.;
      keep col&i.;
   proc transpose data = statistics out = a;
   run;
%mend p_value;
run;
data b (drop = _name_);
set a;
run;
data c;
array coeff[*] col1-col%eval(&num. + 1);
set b;
count = 0;
do i = 2 to &num. + 1;
  if abs(coeff[1]) < abs(coeff[i]) then do;
    count = count + 1;
  end;
end;
empirical_p = count /&num.;
run;
data &dataname._pdataset&x.; set c;
keep empirical_p;
run;
%append (base_data=&dataname._pdataset1, append_data=& dataname._pdataset&x.);
%end;
data &dataname._pdataset;
set &dataname._pdataset1; if _n_ = 1 then delete;
run;
proc transpose data = &dataname._pdataset out = pdataset_new;
  var _all_;
run;
data final_&dataname._pdataset&q.; set pdataset_new; drop _name_;
run;
%mend p_value;
%macro size_of_test;
%do j = 1%to &no_sim_size.;
  %montecarlo_estimation;
  %p_value(q=&j.,dataname=mc_statistics,num=&no_simulation.);
  %append (base_data=final_mc_statistics_pdataset1, append_data=final_ mc_statistics_pdataset&amp;j.);
%end;
data final_mc_pdataset; set final_mc_statistics_pdataset1; if _n_ = 1 then delete;
run;
%macro size_statistics;
  data size_of_test; set sot1;
    if _n_ = 1 then delete;
    five_percent = five_percent_sum/_freq_; keep five_percent;
    label five_percent = "five percent";
run;
proc transpose data = size_of_test out = sizeofetest; var five_percent;
run;
data sizeofest(drop= _name_ ); length _label_ $ 15; set sizeofest;
  rename col1-col48 = t1-t48; rename _level_ = alpha;
run;
%mend size_of_test;
%of test;
(v) Calculation of power of the test:
%macro power_of_test;
  %bootstrap_estimation;
  %bootstrap_data_preparaion;
  %p_value(q=1,dataname=f_boot_pdata,num=&first_bootsample.);
data final_fboot_pdata;
set final_fboot_pdata1;
run;
%do q = 1%to &first_bootsample.;
  %let j = 0;
  %p_value(q=&q., dataname=s_boot_pdata& q.&j.,num= & second_bootsample.);
  %append (base_data=final_s_boot_pdata10_pdataset1, append_data=&syslast);
%end;
data final_sboot_pdata; set final_s_boot_pdata10_pdataset1;
if _n_ = 1 then delete;
run;
data power_dataset; set final_fboot_pdata final_sboot_pdata;
run;
%mend power_of_test;
%macro power_of_test;
data double_boot_pvalue; set final_fboot_pdata1;
run;
proc transpose data = double_boot_pvalue out = power_ calculation;
  var _all_;
run;
data power; set power_calculation; beta = round((1 - col1),0.001);
  keep beta;
run;
proc transpose data = power out = power_of_test; var _all_;
run;
data power_of_test; set power_of_test;
  rename col1-col48 = t1-t48; rename _name_ = power;
  label _name_ = "power of the test";
run;
%mend power_of_test;
%of test;
Pashudhan

Indian Livestock Industry Fair
10-14 December 2014
PUNE, MAHARASHTRA

pashudhan.com