ARIMA Modeling and Forecasting: An Interactive Program Based on IMSL Subroutine Package - II

by

Prem K. Goel* and A. Gregory Rocco

Technical Report #82-12

Department of Statistics

Purdue University

May 1982

*This work was supported in part by the National Science Foundation under grant #MCS-8002336 at Purdue University.

ARIMA Modeling and Forecasting: An Interactive Program Based on IMSL Subroutine Package - II

by

Prem K. Goel and A. Gregory Rocco

This technical report gives the listing of 'arimaf' program. The program description and information for the user is given in Goel, Prem K. and Rocco, A. Gregory (1982) ARIMA Modeling and Forecasting: An Interactive Program Based on IMSL Subroutines Package - I, Technical Report #82-11, which is available from the department on request.

```
c arimaf.f  A. Gregory Rocco  4/29/82
c purdue university
c written for stat 520 for Prof. Prem K. Goel
c
c this program is for arima model fitting using the imsl subroutine package
c
c it is written in standard fortran 77; except for statment #2000 (if system
c call is being used to retrieve the help file).
c
c this program makes use of the single precision imsl library routines
c therefore the command to compile and link this program is as follows:
c     f77 arimaf.f -limslsp
c
c Note: In order for the help command to work properly the path name in the
c open statement at statement #2000 must be correct (unless using system
c call to retrieve help file).
c
c for more info see help file.
c
c               E D I T   H I S T O R Y
c               -------   -------------
c
c  2/ 1/82 v1.08 A. G. Rocco  initial debugged version.
c
c  2/ 3/82 v1.09 A. G. Rocco  split difference command into dif and trans
c
c  2/16/82 v2.00 A. G. Rocco  added the following commands:
c                               out, smodel, rmodel, update
c
c  2/23/82 v2.01 A. G. Rocco  changed filename ('ifname') from a14 format
c                               to a60 to allow for path names.
c  3/20/82 v2.01 A. G. Rocco  made minor corrections to  comments.
c
c  3/28/82 v2.02 A. G. Rocco  if error while finding preliminary estimates
c                               for arima model, set the preliminary estimates
c                               to .1. Fixed error which caused last page
c                               of plot to be droped if it only had 1 point.
c
```

```
c  4/26/82 v2.03 A. G. Rocco  put in pathname for help file.  added "?"
c                               command (same as help). corrected file error
c                               routines (added close file commands among
c                               other things).  made minor changes to
c                               comments.  changed unknown command message.
c
c  4/29/82 v2.04 A. G. Rocco  changed help routine to use system call;
c                               commented out old routine, in case need later.
c
c
c   Note: if the length of 'ifname' is changed 'cblank' must also be changed!
        character icom*3,icom1*1,icom2*2,iworkc(18)*4,ifname*60
        character digits(0:11)*1, ctemp(118)*1, cblank*14
c
        logical ifdout,iftout,ifdev(3),ltemp,fmodel,fagain
        logical flargs,farerr
c
        dimension orgdat(500),data(500),res(1000),work1(150)
        dimension var(50),acv(50),fcst(3,250),ac(50),pacv(50)
        dimension iwork1(118),work2(500)
        dimension arps(10),pmas(10)
        dimension idev(3)
c
c
10      format(' arimaf version 2.04')
c
c
c single varibles and constants
c
c itin - terminal input unit number
c itout - terminal output unit number
c idout - disk output unit number
c iofile - unit number for other disk i/o
        data itin/5/,itout/6/,iofile/1/,idout/2/
c
c mdata - maximum number of data points
c mac   - max number of autocorrelations also max # of partial autocorrelations
c mplotw - max width of plot (horizontal)
c iplotw - plot width (horizontal)
c iplotl - number of lines to plot
c iper - period of plot routine
        data mdata/500/,mac/50/,iper/10/
        data mplotw/118/,iplotw/66/,iplotl/19/
        data (digits(i),i=0,11)/'1','2','3','4','5',
     +'6','7','8','9','0','A','B'/
c
c cblank - 60 blanks to see if no filename input
        data cblank
     +/'                                                            '/
c
c
c nac is the number of acf, pacf and acv points calculated.
c marma - is max order of ar and ma models (separately).
c mfcst is max lead time for forecasting.
        data marma/10/,mfcst/250/
```

```
c
c miter - is max number of iterations for arima fit
c nsig - is number of significant digits of the objective function which do
c  not change after 'niter' iterations of arma fit.
c
        data miter/25/,nsig/6/,niter/3/
c
c ifdout - when true enables output to disk
c iftout - when true enables long output to terminal
c fmodel - when true an arima model has been found
c
c array varibles these arrays must be dimensioned to
c be at least the value of the statements listed under them.
c
c orgdat - contains the original data read from the file
c       1. mdata
c
c data  - array being analyzed
c       1. mdata
c
c res  - residuals stored here also used as working storage for ftmxl
c       1. (marma+6) + marma +1
c       2. ((marma+3)*marma)/2+6*marma+marma+6
c       3. 2*mdata
c
c work1 - working storage, used by ftauto, ftarps, ftmps, ftmxl
c   also used as bit bucket for ftcast.
c       1. mac
c       2. marma**2+5*marma
c       3. (5*(marma+marma)
c       4. 5*(marma+1))+(((marma+3)*marma)/2
c       5. marma+ # of differences
c
c work2 - used to input data to ftdif and ftmxl
c       1. mdata
c
c ctemp - used to buffer line to be plotted
c       1. mplotw
c       2. 118
c
c ifname - used to read in file names
c
c iwork1 - used by plot routine, ftcast and ftmxl
c       1. mplotw
c       2. 7
c       3. 24
c
c arps - ar parameter est.
c       1. marma
c
c pmas - ma paramater est.
c       1. marma
c
c
c fcst - used to store forecast
c       1. 3 x mfcst
c
```

```
c var - used to store variance after calculated by ftauto
c       1. mac
c
c ac - autocorrelations
c       1. mac
c
c pacv - partial autocorrelations
c       1. mac
c
c ifdev(1) contains a flag to indicate if outputting to the terminal or not
c ifdev(2) contains a flag to indicate if outputting to disk
c ifdev(3) is always true for output which is to go to the terminal regardless
c   of ifdev(1)
c
c idev(1), 2, 3 contains unit numbers for terminal, disk, terminal respectively
c
        idev(1)=itout
        idev(2)=idout
        idev(3)=itout
c
c format statements used by several different routines
c   filenames read in a60 format to allow for path name.
800     format(a60)
801     format(a2)
810     format(f16.5)
811     format(i9)
c
820     format(' filename? ',$)
822     format(' please answer on or off')
823     format(' error in writing output file')
824     format(' ')
825     format('

                ')
c
c output version number and initilize things
c
        write(itout,10)
        norg=0
        ndata=0
        nres=0
        nac=0
        nfor=0
        nwork2=0
        iupst=0
c
        ifdout=.false.
        iftout=.true.
        ifdev(1)=.true.
        ifdev(2)=.false.
        ifdev(3)=.true.
c
        fmodel=.false.
c
c command decode routine
c
```

```
900     format(' arm>',$)
910     format(1a3,19a4)
920     format(' Unknown command; type help for command summary.')
c
c
        goto 1000
c
c error return - before getting next command increment error count so
c   if executing update commmand can take appropriate action
c
990     iecnt=iecnt+1
        goto 1005
c
c normal return set count of consecutive errors to zero
1000    iecnt=0
c
c output a blank line and if an update command is in progress
c then return to update routine
c
1005    write(itout,824)
        if (iupst.ne.0) goto 2700
        iecnt=0
c
        write(itout,900)
        read(itin,910,err=1050) icom,(iworkc(i),i=1,19)
c
c output command to disk if it is enabled
c after stripping trailing blanks
        if (ifdout) then
           ilen=19
c
1010       if(iworkc(ilen).eq.'    ') then
              ilen=ilen-1
              goto 1010
           end if
c
           write(idout,824)
           write(idout,'(1x,1a3,19a4)') icom,(iworkc(i),i=1,ilen)
        end if
c
        if(icom.eq.'not') goto 1000
        if(icom.eq.'inp') goto 1200
        if(icom.eq.'ope') goto 1300
        if(icom.eq.'clo') goto 1400
        if(icom.eq.'fil') goto 1500
        if(icom.eq.'ter') goto 1600
        if(icom.eq.'pri') goto 1700

        if(icom.eq.'swa') goto 1800
        if(icom.eq.'ori') goto 1900
        if(icom.eq.'hel') goto 2000
        if(icom.eq.'?  ') goto 2000
        if(icom.eq.'sta') goto 2200
c
```

```
        if(icom.eq.'dif') goto 2300
        if(icom.eq.'tra') goto 2325
        if(icom.eq.'out') goto 2400
        if(icom.eq.'smo') goto 2500
        if(icom.eq.'rmo') goto 2600
        if(icom.eq.'upd') goto 2700
c
        if(icom.eq.'acf') goto 3000
        if(icom.eq.'pac') goto 3100
c
        if(icom.eq.'ari') goto 4000
        if(icom.eq.'aga') goto 4020
        if(icom.eq.'pwi') goto 4800
        if(icom.eq.'ple') goto 4850
        if(icom.eq.'plo') goto 5000
        if(icom.eq.'for') goto 6000
        if(icom.eq.'con') goto 6800
        if(icom.eq.'exi') goto 10000
c
        write(itout,920)
        goto 1000
c
c here on illegal input.  flush input buffer then output error message.
1050    read(itin,'(118a1)') (ctemp(i),i=1,118)
        write(itout,'('' error on input probably due to illegal '',
     +    ''input'')')
        goto 990
c
cccccccccccccccccccccccccccccc
c
c routine for inputting a file
c
1191    format(' ',i7,' data points read from file: ',a60)
1192    format(' ',f16.5,3(2x,f16.5),' . . .')
1193    format(' file error trying to read: ', 1a60)
1194    format(' warning file too big . . . did not reach end of file')
c
1200    write(itout,'('' input data from file? '',$)')
        read(itin,800,err=1050) ifname
c
c if no filename input, then exit
        if (ifname.eq.cblank)  goto 1000
c
        open(iofile,file=ifname,status='old',
     +    err=1275,access='sequential')
        rewind(unit=iofile,err=1275)
c
c set number of auto correlations already calculated to zero and init counter
        nac=0
c
        i=1
1210       if (i.le.mdata) goto 1220
c
              write(itout,1194)
              goto 1250
```

```
c
1220          read(iofile,*,end=1250,err=1285) orgdat(i)
              data(i)=orgdat(i)
              i=i+1
              goto 1210
c
1250       close(iofile,err=1290)
           norg=i-1
           ndata=norg
c
           write(itout,1191) norg,ifname
c
           if (norg.ge.4)  then
              itemp=4
           else
              itemp=norg
           end if
c
           write(itout,1192) (orgdat(i),i=1,itemp)
c
           if (ifdout) then
              write(idout,1191) norg,ifname
              write(idout,1192) (orgdat(i),i=1,itemp)
           end if
c
           goto 1000
c
c these error routines are shared by serveral of the disk i/o routines
c
c error opening file
1275       write(itout,'('' error opening file: '',a60)') ifname
           close(iofile,err=990)
           goto 990
c
c error reading file
1285       write(itout,1193) ifname
           close(iofile,err=990)
           goto 990
c
c error writting file
1288       write(itout,'('' error writting to file: '',a60)') ifname
           close(iofile,err=990)
           goto 990
c
c error closing
1290       write(itout,'('' error closing file: '',a60)') ifname
           goto 990
c
ccccccccccccccccccccccccccccccc
c
c routine for opening an output file
c
```

```
1300    write(itout,820)
        read(itin,800,err=1050) ifname
        if (ifname.eq.cblank)  goto 1000
        nac=0
        npacv=0
c
        open(idout,file=ifname,err=1375,access='sequential')
        ifdout=.true.
        ifdev(2)=ifdout
        goto 1000
c
c here if error opening output file
1375    write(itout,'('' error opening output file: '',a60)') ifname
        close(idout,err=990)
        goto 990
c
ccccccccccccccccccccccccccccccc
c
c routine for closing output file
c
1390    format(' error closing output file')
c
1400    ifdout=.false.
        ifdev(2)=ifdout
        close(idout,err=1450)
        goto 1000
c
c error closing output file
1450    write(itout,1390)
        goto 990
c
ccccccccccccccccccccccccccccccc
c
c file command for specifying whether file output should be on or off
c
1490    format(' do you want file output on or off? ',$)
c
1500    write(itout,1490)
        read(itin,801,err=1050) icom2
c
        if (icom2.eq.'on') then
           ifdout=.true.
        else
           if (icom2.eq.'of') then
              ifdout=.false.
           else
c
c error
              write(itout,822)
              goto 1500
           end if
        end if
        ifdev(2)=ifdout
        goto 1000
c
```

```
cccccccccccccccccccccccccccccc
c
c terminal command for specifying whether long terminal output is enabled.
c
1590    format(' do you want the output of tables and graphs ',
     +    'to the terminal on or off? ',$)
c
1600    write(itout,1590)
        read(itin,801,err=1050) icom2
c
        if (icom2.eq.'on') then
           iftout=.true.
        else
           if (icom2.eq.'of') then
              iftout=.false.
           else
c
c error
              write(itout,822)
              goto 1600
           end if
        end if
        ifdev(1)=iftout
        goto 990
c
cccccccccccccccccccccccccccccc
c
c this routine prints out data in the array data
c
1690    format(1x,f16.5,3(2x,f16.5))
c
c
1700    do 1740 i4=1,2
           if (ifdev(i4)) then
              iodev=idev(i4)
              write(iodev,'(1x,i5,'' data points'')') ndata
c
              do 1730 i2=1,ndata-4,4
                 write(iodev,1690) (data(i),i=i2,i2+3)
1730          continue
              write(iodev,1690) (data(i),i=i2,ndata)
c
           end if
1740    continue
c
        goto 1000
c
c here if error
1775    write(itout,823)
        goto 990
c
cccccccccccccccccccccccccccccc
c
c this routine swaps the residuals with the data so they can be analyzed.
c
```

```
1800    nac=0
        if (nres.gt.ndata) then
           n=nres
        else
           n=ndata
        end if
        do 1810 i=1,n
           temp=data(i)
           data(i)=res(i)
           res(i)=temp
1810    continue
c
        itemp=ndata
        ndata=nres
        nres=itemp
        goto 1000
c
cccccccccccccccccccccccccccccc
c
c   this routine moves original data back into data column for further analysis
c
1900    nac=0
        npacv=0
        do 1910 i=1,norg
           data(i)=orgdat(i)
1910    continue
        ndata=norg
        goto 1000
c
cccccccccccccccccccccccccccccc
c
c help routine ... copies the file specified to the terminal, or calls
c system help routine. Note: this system call is non-standard F77.
c the inappropriate routine is commented out with 'cx 's.
c
ccccccccccc
c
c help routine using system call.
c
2000    call system('help arimaf')
        goto 1000
c
ccccccccccc
c
cx c help routine which copies the help file specified by the path name given
cx c in statement #2000 to the terminal.
cx c
cx 1990    format(1x,20a4)
cx 1995    format(20a4)
cx c
cx c enter here
cx 2000    open(iofile,file='/usr/man/man1/arimaf.1',status='old'
cx      +   ,err=2175,access='sequential')
cx         rewind(unit=iofile,err=2175)
cx c
cx 2040    read(iofile,1995,err=2175,end=2060) (iworkc(i),i=1,20)
cx
cx c
cx c don't print trailing blanks
cx            itemp1=20
cx
cx 2050       if (iworkc(itemp1).eq.'    ') then
cx               itemp1=itemp1-1
cx               goto 2050
cx            end if
cx c
cx            write(itout,1990) (iworkc(i),i=1,itemp1)
cx            goto 2040
cx c
cx c here on end of file
cx 2060    close(iofile,err=1000)
cx         goto 1000
cx c
cx c here on error
cx 2175    write(itout,'('' error reading help file'')')
cx         close(iofile,err=2175)
cx         goto 990
c
cccccccccccccccccccccccccccc
c
c output status of columns

c
2200    do 2250 i=1,2
           if(ifdev(i)) then
              iodev=idev(i)
c
              write(iodev,824)
              write(iodev,'(1x,i5,'' data points in column '',
     +          ''of original data.'')') norg
              write(iodev,'(1x,i5,'' data points in column '',
     +          ''to be analyzed.'')') ndata
              write(iodev,'(1x,i5,'' data points in column '',
     +          ''of residuals.'')') nres
              write(iodev,'(1x,i5,'' forecasts in forecast '',
     +          ''array.'')') nfor
           end if

2250    continue
        goto 1000
c
ccccccccccccccccccccccccccc
c
c routine to difference or transform the data in the data column.
c this routine has two different entry addresses depending on whether
c the data is to be differenced or transformed.
c
2299    format(' warning this routine will destroy the contents of ',
     +   'the data column.')
c
c
```

```
2300       ip=1
           write(itout,2299)
           write(itout,'('' order of'',
     +       '' non-seasonal difference? '',$)')
           read(itin,811,err=1050) idif
c
c if number of differences neg abort.
           if (idif.lt.0)  goto 990
c
           write(itout,'('' order of seasonal difference? '',$)')
           read(itin,811,err=1050) isord
           if (isord.lt.0)  goto 990
c
           if (isord.gt.0)  then
              write(itout,'('' length of seasonal period? '',$)')
              read(itin,811,err=1050) isper
              if (isper.lt.0)  goto 990
           end if
        goto 2335
c
c enter here for transformation
2325       write(itout,2299)
           idif=0
           isord=0
c
           write(itout,'('' power transformation exponent ''
     +       ''(or enter 0 if log)? '',$)')
           read(itin,811,err=1050) ip
c
c clear number of acfs and pacfs calculated to zero
2335    nac=0
        call ftrdif(idif,isord,ip,isper,ndata,data,shift,lw,ierr)
        ndata=lw
c
        do 2350 i=2,3
           if (ifdev(i)) then
              iodev=idev(i)
              write(iodev,824)
c
              if (ip.ne.1) then
                 if (ip.eq.0) then
                    write(iodev,'('' performing natural log '',
     +                ''transformation: data(i)=log(data(i) +'',f16.5,
     +                '' )'')') shift
                 else
                    write(iodev,'('' performing exponential '',
     +                ''transformation: data(i)=data(i) **'',i9)') ip
                 end if
              end if
c
              if (idif.gt.0) then
                 write(iodev,'('' finding non-seasonal difference '',
     +             ''of order '',i4)') idif
              end if
              if (isord.gt.0)  then
                 write(iodev,'('' finding seasonal difference of order '',
     +             i4,'' with period '',i4)') isord,isper
              end if
c
              if (ierr.ne.0) then
                 write(iodev,'('' error= '',i4,'' while finding'',
     +            '' difference or transformation.'')') ierr
              end if
           end if
2350    continue
        goto 1000
c
cccccccccccccccccccccccc
c
c routine for outputting data column to a file
c
2390    format(1x,i7,' data points output to file: ',a60)
c
2400    write(itout,'('' output data to file? '',$)')
        read(itin,800,err=1050) ifname
        if (ifname.eq.cblank)  goto 1000
c
        open(iofile,file=ifname,err=1275,access='sequential')
        rewind(unit=iofile,err=1275)
c
c output data
        do 2420 i=1,ndata
           write(iofile,*,err=1288) data(i)
2420    continue
c
        close(iofile,err=1290)
c
        write(itout,2390) ndata,ifname
c
        if(ifdout) then
           write(idout,2390) ndata,ifname
        end if
        goto 1000
c
cccccccccccccccccccccccc
c
c routine for saving a model
c
2490    format(' model written to file: ',a60)
c
2500    write(itout,'('' write model to file? '',$)')
        read(itin,800,err=1050) ifname
        if (ifname.eq.cblank) goto 1000
c
        open(iofile,file=ifname,err=1275,access='sequential')
        rewind(unit=iofile,err=1275)
c
```

```
c ouput model to disk
        write(iofile,*,err=1288)  iar,idif,ima,ndpts
        write(iofile,*,err=1288)  pmac,wnv
c
        if (iar.gt.0)  then
           write(iofile,*,err=1288)  (arps(i),i=1,iar)
        end if
c
        if (ima.gt.0)  then
           write(iofile,*,err=1288)  (pmas(i),i=1,ima)
        end if
c
        close(iofile,err=1290)
c
        write(itout,2490) ifname
        if (ifdout) then
           write(idout,2490) ifname
        end if
c
        goto 1000
c
cccccccccccccccccccccccc
c
c routine for retrieving a model
c
2590    format(' arima model retrieved from file: ',a60)
c
2600    write(itout,'('' retrieve model from file? '',$)')
        read(itin,800,err=1050) ifname
        if (ifname.eq.cblank) goto 1000
c
        open(iofile,file=ifname,status='old',
     +    err=1275,access='sequential')
        rewind(unit=iofile,err=1275)
c
        read(iofile,*,err=1285,end=1285)  iar,idif,ima,ndpts
        read(iofile,*,err=1285,end=1285)  pmac,wnv
c
        if (iar.gt.0) then
           read(iofile,*,err=1285,end=1285)  (arps(i),i=1,iar)
        end if
c
        if (ima.gt.0) then
           read(iofile,*,err=1285,end=1285)  (pmas(i),i=1,ima)
        end if
c
        close(iofile,err=1290)
c
        write(itout,2590) ifname
        if (ifdout)  then
           write(idout,2590)  ifname
        end if
c
c go output model to enabled devices.
        goto 4450
c
```

```
cccccccccccccccccccccccccccccc
c
c routine for updating a forecast.
c iupst is used to keep tract of which state this routine is in. after
c each piece of this routine is executed, control is returned here via
c the normal command reentry address ( just before the commnad decode
c table).  iupst can take the following values:
c
c  0  update not in progress
c  1  retrieve input file
c  2  retrieve model
c  3  add new point
c  4  make new forecast
c  5  output new data file
c
c if an error occurs during any of the above states (except 0), retry
c three time before returning to command mode.
c
c
c increment state count, if no errors
2700    if (iecnt.eq.0)   iupst=iupst+1
c
c have 3 consecutive errors occurred
        if (iecnt.ge.3)  then
           iupst=0
           goto 1000
        end if
c
c goto appropriate routine or exit.
        goto (1200,2600,2750,6000,2400,2780) iupst
c
c routine to add new observation
2750    write(itout,'('' new observation to be added to data? '',$)')
        read(itin,*,err=1050) temp
        ndata=ndata+1
        data(ndata)=temp
        goto 1000
c
c here when finished updating model
2780    iupst=0
        goto 1000
c
ccccccccccccccccccc
c
c routines for calculating acf, pacf and acv will use variable nac
c to tell if they actually need to calculate more terms.
c nac - is the number of acf, pacf, and acv terms already calculated.
c
c routine for finding acf
c
2990    format(' number of terms? ',$)
2991    format(' to big max value is: ',i6)
c
c input number of acf terms to be output
3000    write(itout,2990)
        read(itin,811,err=1050) nacp
c
```

```
c if no input default
        temp=ndata
        if (nacp.eq.0)  nacp=sqrt(temp)+10
c
c make sure input is legal
        if (nacp.gt.mac) then
c
c here if error
           write(itout,2991) mac
           goto 3000
        end if
c
c calculate acf if need to  note: calculate pacf also in case it is needed
c later.
c

        if (nac.lt.nacp) then
           call ftauto(data,ndata,nacp,nacp,7,
     +      amean,var,acv,ac,pacv,work1)
           nac=nacp
        end if
c
c plot acf
        if(iftout)   call plot1(itout,'acf ',nacp,ac,amean,var)
        if(ifdout)   call plot1(idout,'acf ',nacp,ac,amean,var)
        goto 1000
c
c routine for finding pacf
c
c input number of pacf terms to be output
3100    write(itout,2990)
        read(itin,811,err=1050) npacvp
c
c if no input default
        temp=ndata
        if (npacvp.eq.0) npacvp=sqrt(temp)+10
c
c make sure input is legal
        if (npacvp.gt.mac) then
c here if error
           write(itout,2991) mac
           goto 3100
        end if
c
c calculate pacf if need to then output
        if (nac.lt.npacvp) then
           call ftauto(data,ndata,npacvp,npacvp,7,amean,var,
     +       acv,ac,pacv,work1)
           nac=npacvp
           if (nac.lt.npacvp)  nac=npacvp
        end if
c
c plot pacf on enable output devices
3280    if (iftout)   call plot1(itout,'pacf',npacvp,pacv,amean,var)
        if  (ifdout)   call plot1(idout,'pacf',npacvp,pacv,amean,var)
        goto 1000
c
```

```
ccccccccccccccccccccccccccccc
c
c arima fitting routine
c
3900    format(' input desired arima model? ',$)
3905    format(3i4)
3910    format(' illegal model')
3915    format(' would you like to input preliminary estimates? ',$)
3920    format(1a1)
3925    format(' preliminary estimates for ar model? ',$)
3930    format(' preliminary estimate for ma model? ',$)
3940    format(' error=',i4,' while taking differences')
3945    format(' error=',i4,' while '
     + 'finding preliminary est. for AR model .1(s) assumed.')
3950    format(' error=',i4,' while '
     + 'finding preliminary est. for MA model .1(s) assumed.')
3951    format(' initial ar parameters:')
3952    format(6x,f9.5,4(5x,f9.5))
3955    format(' initial ma parameters:')
3958    format(' warning after ',i5,
     + ' iterations the model parameters did not converge')
3960    format(' error=',i4,' while doing arma fit using ftmxl')
c
3970    format(' max likelihood est.',
     + 'of arima model (',2(i2,','),i2,')')
3972    format(' for ',i6,' data points after differencing')
3975    format(4x,'AR terms:')
3980    format(4x,'MA terms:')
c
c enter here for first time fit
c clear again flag and goto begining of routine.
4000    fagain=.false.
        goto 4030
c
c enter here if running fit again
c set flag so will skip sections of code which should not be reexecuted.
4020    fagain=.true.
c
c make sure there is input data
4030    if(ndata.eq.0) then
           write(itout,'('' no data'')')
           goto 990
        end if
c
c copy data into work2
        do 4050 i=1,ndata
           work2(i)=data(i)
4050    continue
c
        nwork2=ndata
c
        if (.not.fagain)  then
c
c input desired model
           write(itout,3900)
           read(itin,*,err=1050) iar,idif,ima
        end if
c
```

```
c see if parameters ok
        if ((iar.lt.0).or.(iar.gt.marma).or.(idif.lt.0).or.
     +   (ima.lt.0).or.(ima.gt.marma)
     +   .or.((ima.eq.0).and.(iar.eq.0)))  then
           write(itout,3910)
           goto 4000
        end if
c
c perform differencing operation
        if (idif.gt.0) then
           call ftrdif(idif,0,1,0,nwork2,work2,0,lw,ierr)
           nwork2=lw
           if (ierr.ne.0) then
               write(itout,3940) ierr
               if (ifdout)  then
                  write(idout,3940)  ierr
               end if
           end if
        end if
c
c input preliminary est. if none calculate.
4100    if (.not.fagain) then
           write(itout,3915)
           read(itin,3920,err=1050) icom1
           if (icom1.eq.'y') then
              ltemp=.true.
           else
              if (icom1.eq.'n') then
                 ltemp=.false.
              else
                 write(itout,'('' Please answer yes or no'')')
                 goto 4100
              end if
           end if
c
           if (ltemp) then
              if (iar.ge.1) then
                 write(itout,3925)
                 read(itin,*,err=1050) (arps(i),i=1,iar)
              end if
c
              if (ima.ge.1) then
                 write(itout,3930)
                 read(itin,*,err=1050) (pmas(i),i=1,ima)
              end if
c
           else
c
c calculate preliminary estimates by:
c 1. find autocovariances and mean of data to be fit
c 2. using ftarps to get est for ar
c 3. use ftmps to get est for ma paramaters
c note: error messages will be output to disk if enabled.
c
```

```
              iordr1=iar+ima+1
              nac=0
              call ftauto(work2,nwork2,iordr1,0,3,amean,
     +         acv(1),acv(2),ac,pacv,work1)
c
c
c find est for ar terms
c farerr and ierr are used after call to ftmps to see if ftarps ran ok
              farerr=.false.
              ierr=0
              if (iar.ge.1) then
                 call ftarps(acv,amean,iar,ima,arps,
     +             pmac,work1,ierr)
c
                 if (ierr.ne.0) then
                    farerr=.true.
                    do 4190 i2=1,iar
                       arps(i2)=.1
4190                continue
c
                    do 4200 i=2,3
                       if (ifdev(i)) then
                          iodev=idev(i)
                          write(iodev,3945) ierr
                          if (ierr.eq.129) then
                             write(iodev,'('' error 129 above '',
     +                       ''is probably due nonstationarity.'')')
                          end if
                       end if
4200                continue
                 end if
c
              end if
c
c find est for ma terms
c if there was an error calculating preliminary AR estimates, don't call ftmps
c
              if (ima.ge.1) then
                 if (.not.farerr ) then
                    call ftmps(acv,arps,iar,ima,pmas,wnv,work1,ierr)
                 end if
c
                 if (ierr.ne.0) then
                    do 4290 i2=1,ima
                       pmas(i2)=.1
4290                continue
c
                    do 4300 i=2,3
                       if (ifdev(i)) then
                          iodev=idev(i)
                          if (.not.farerr) then
                             write(iodev,3950) ierr
                          end if
                       end if
```

```
4300                continue
                 end if
c
              end if
           end if
        end if
c
c output preliminary estimates of arma parameters to terminal (itout)
c regardless of whether it is enabled or not and output to the disk (idout)
c if the flag is on.  find the model then output model on same devices.
c
c only output for models being fit
c
        do 4400 i=2,3
           if (ifdev(i)) then
              iodev=idev(i)
c
              if (iar.gt.0) then
                 write(iodev,824)
                 write(iodev,3951)
                 write(iodev,3952) (arps(i2),i2=1,iar)
              end if
c
              if (ima.gt.0) then
                 write(iodev,824)
                 write(iodev,3955)
                 write(iodev,3952) (pmas(i2),i2=1,ima)
              end if
c
           end if
4400    continue
c
c
c set up and perform arma fit
c
        iwork1(1)=nwork2
        iwork1(2)=iar
        iwork1(3)=ima
        iwork1(4)=0
        iwork1(5)=miter
        iwork1(6)=nsig
        iwork1(7)=1
        iwork1(8)=niter
c
        ndpts=nwork2
c
        call ftmxl(work2,iwork1,arps,pmas,pmac,wnv,work1,res,ierr)
c
c enter here to output model after retrieving it from disk.
c output fit
4450    do 4500 i1=2,3
           if (ifdev(i1)) then
              iodev=idev(i1)
c
              if (ierr.eq.68) then
                 write(iodev,3958) miter
              else
```

```
                 if (ierr.ne.0) then
                    write(itout,3960) ierr
                 end if
              end if
c
              write(iodev,824)
              write(iodev,3970) iar,idif,ima
              write(iodev,3972) ndpts
              write(iodev,'(4x,'' est. of moving average '',
     +          ''constant = '',f16.5)') pmac
              write(iodev,'(4x,'' est. of white noise '',
     +          ''variance = '',f16.5)') wnv
              write(iodev,824)
              if (iar.gt.0) then
                 write(iodev,3975)
                 write(iodev,3952) (arps(i),i=1,iar)
                 write(iodev,824)
              end if
c
              if (ima.gt.0) then
                 write(iodev,3980)
                 write(iodev,3952) (pmas(i),i=1,ima)
                 write(iodev,824)
              end if
           end if
4500    continue
c
        nres=nwork2
        fmodel=.true.
        goto 1000
c
cccccccccccccccccccccccccccc
c
c input plot width
c
4800  write(itout,'('' plot width? '',$)')
      read(itin,*,err=1050) iplotw
c
      if ((iplotw.le.0).or.(iplotw.gt.mplotw)) then
         write(itout,'('' must be between 1 and '',i5)') mplotw
         goto 4800
      end if
      goto 1000
c
c input plot length
c
4850  write(itout,'('' plot length? '',$)')
      read(itin,*,err=1050) iplotl
c
      if (iplotl.le.1) then
         write(itout,'('' must be greater than 1 '')')
         goto 4850
      end if
      goto 1000
c
```

```
ccccccccccccccccccccccccccccccc
c
c routine for plotting data
c
c
c input period
5000  write(itout,'('' plot period?'',$)')
      read(itin,811,err=1050) itemp1
c
      if ((itemp1.lt.0).or.(itemp1.gt.12)) then
         write(iout,'('' period must be between 1 '',
     +     ''and 12'')')
         goto 5000
      end if
c
c if input equal to zero don't change period
      if (itemp1.ne.0) then
         iper=itemp1
      end if
c
c find max and min
        fmax=data(1)
        fmin=data(1)
c
        do 5100 i=2,ndata
           if (data(i).gt.fmax) then
              fmax=data(i)
           end if
c
           if (data(i).lt.fmin) then
              fmin=data(i)
           end if
5100    continue
c
c calculate scale factor which is change in y for a change of one line
c if scale is going to be zero fix it, so won't divide by 0
        temp=fmax-fmin
        if (temp.lt..0001) then
           if (fmax.eq.0)  then
              fmax=.0001
              fmin=-.0001
           else
              temp2=1.1
              if (fmax.lt.0) then
                 temp2=1/temp2
              end if
              fmax=fmax*temp2
              fmin=fmin/temp2
           end if
        end if
c
        scale=(fmax-fmin)/(iplotl-1)
        ntoplt=ndata
        ipoint=1
c
```

```
c pick proper output format for x axis
      temp2=abs(fmax)
      temp1=abs(fmin)
c
      if (temp1.gt.temp2)  then
         temp2=temp1
      end if
c
      if (temp2.gt.9999.999) then
         flargs=.true.
      else
         flargs=.false.
      end if
c
c keep looping back to here to plot another page
5200  if (ipoint.le.ndata) then
         istart=ipoint
         if (iftout) then
            write(itout,825)
         end if
c
         if (ifdout) then
            write(idout,825)
         end if
c
         if (ntoplt.gt.iplotw) then
            npts=iplotw
         else
            npts=ntoplt
         end if
c
c scale points to be plotted
         do 5300 i=1,npts
            iwork1(i)=((data(ipoint)-fmin)/scale)+.5
            ipoint=ipoint+1
5300     continue
c
c plot out iplotl lines to whichever devices are enabled
         do 5600 i1=1,2
            if (ifdev(i1)) then
               iodev=idev(i1)
c
c print each line
               do 5500 i2=(iplotl-1),0,-1
                  ipoint=ipoint-npts
c
                  do 5400 i3=1,npts
c
c decide whether to output a point or a blank one space at a time,
c fill buffer and output it.
                     if (iwork1(i3).eq.i2) then
                        itemp1=(ipoint-1)/iper
                        itemp2=ipoint-1-itemp1*iper
                        ctemp(i3)=digits(itemp2)
                     else
                        ctemp(i3)=' '
                     end if
c
```

```
                       ipoint=ipoint+1
5400                continue
c
c remove trailing blanks and output line
                    i4=npts
5450                if (ctemp(i4).ne.' ') then
                       if (flargs) then
                          write(iodev,'(1x,f11.0,1x,119a1)')
     +                      fmin+i2*scale,(ctemp(i5),i5=1,i4)
                       else
                          write(iodev,'(1x,f11.5,1x,119a1)')
     +                       fmin+i2*scale,(ctemp(i5),i5=1,i4)
                       end if
                    else
                       i4=i4-1
                       goto 5450
                    end if
c
5500             continue
c
c output axis along the bottom of plot
                 ipoint=ipoint-npts
                 do 5540 i2=1,npts
                    itemp1=ipoint/10
                    itemp1=itemp1*10
c
                    if (ipoint.eq.itemp1) then
                       ctemp(i2)='^'
                    else
                       ctemp(i2)='-'
                    end if
c
                    ipoint=ipoint+1
c
5540             continue
                 write(iodev,'(13x,119a1)')
     +             (ctemp(i2),i2=1,npts)
c
c label axis
                 write(iodev,'(12x'' '',$)')
                 itemp1=ipoint+1-npts
c
5560             if (itemp1.le.(npts+istart)) then
                    itemp2=itemp1/10
                    itemp2=itemp2*10
c
                    if (itemp1.eq.itemp2) then
                       write(iodev,'(i3,$)') itemp1
                       itemp1=itemp1+3
                    else
                       write(iodev,'('' '',$)')
                       itemp1=itemp1+1
                    end if
c
```

```
                   goto 5560
                end if
             end if
5600      continue
c
c update number of points to plot, loop back encase there are more pages
          ntoplt=ntoplt-npts
          goto 5200
       end if
c
       goto 1000
c
cccccccccccccccccccccccccccccc
c
c forecast routine
c
5910   format(3x,' lead',17x,'psi',30x,'std. error')
5920   format(3x,' time',13x,'weights',13x,'forecast',11x,f7.2,' %')
5930   format(3x,' ----',13x,'-------',13x,'-------',12x,'--------')
c
c make sure there is data and a model
6000   if ((ndata.eq.0).or.(.not.fmodel))  then
          write(itout,'(''error either no model or no data'')')
          goto 990
       end if
c
c input alpha
       write(itout,'('' forecast; alpha?'',$)')
       read(itin,810,err=1050) alpha
c
       if ((alpha.lt.0).or.(alpha.gt.1)) then
          write(itout,'('' alpha must be between 0 and 1'')')
          goto 6000
       end if
c
c deefault alpha is .05
       if (alpha.eq.0) then
          alpha=.05
       end if
c
c input lead time
6100   write(itout,'('' lead time? '',$)')
       read(itin,811,err=1050) ileadt
       if (ileadt.eq.0) goto 1000
c
       if (ileadt.le.mfcst) goto 6150
          write(itout,'('' lead time must be less than '',i5)') mfcst
          goto 6100
6150   continue
c
       iwork1(1)=ndata
       iwork1(2)=iar
       iwork1(3)=ima
       iwork1(4)=idif
       iwork1(5)=ileadt
       write(itout,824)
c
```

```
c make forecast
        call ftcast(data,arps,pmas,pmac,alpha,iwork1,work1,
     +    fcst,wnv,ierr)
c
c output error message if any
        if (ierr.ne.0) then
           write(itout,'(''error '',i3,'' while finding '',
     +       ''forecast'')') ierr
           nfor=0
           goto 990
        end if
c
c output forecast
        do 6500 i=1,2
           if (ifdev(i)) then
              iodev=idev(i)
              write(iodev,5910)
              write(iodev,5920) 100*alpha
              write(iodev,5930)
c
              do 6400 ilead=1,ileadt
                 write(iodev,'(1x,i7,3(4x,f16.5))')
     +            ilead,(fcst(i3,ilead),i3=1,3)
6400          continue
c
           end if
c
6500    continue
        nfor=ileadt
        goto 1000
c
ccccccccccccccccccccc
c
c concactinate forecasted points to data
c
6790    format(' lead time truncated to ',i5)
c
6800    itemp1=ndata+nfor
        if (itemp1.gt.mdata) then
           itemp1=mdata
           itemp2=itemp1-ndata
           write(itout,6790) itemp2
c
           if (ifdout) then
              write(idout,6790) itemp2
           end if
        else
           itemp2=itemp1-ndata
        end if
c
        i=ndata
        do 6850 ilead=1,itemp2
           i=i+1
           data(i)=fcst(2,ilead)
6850    continue
c
```

```
        ndata=itemp1
        goto 1000
c
cccccccccccccccccccccccccccccc
c
c exit
c
c if output file open close
10000   if (ifdout) then
           close(idout,err=10075)
        end if
        stop
c
10075   write(itout,'('' error in closing output file'')')
        stop
        end
c
ccccccccccccccccc
c
c subroutine for plotting acf and pacf
c
        subroutine plot1(idev,ititle,npts,stuff,amean,var)
c
        character linbuf(-25:25)*1
        dimension stuff(npts)
        logical ineg
c
c output axis
        write(idev,'(''

                       '')')
        write(idev,'('' time series  mean= '',f16.5,
     +    ''  and  variance= '',f16.5,)') amean,var
        write(idev,'(1x,a4)') ititle
        write(idev,'(13x,''-1.0 -0.8 -0.6 -0.4 -0.2  0.0'',
     + ''  0.2  0.4  0.6  0.8  1.0'')')
        write(idev,'(15x,''+'',10(''----+''))')
c
c output each point
        do 2000 ipoint=1,npts
c
c calculate where point goes on a scale of -25 to +25
           value=stuff(ipoint)+1
           iplot=25*value+.5
           iplot=iplot-25
c
c make up line buffer by going through each position and decide if ' ' or 'x'
           ineg=(iplot.lt.0)
           do 1500 i=-25,25
              if (ineg) then
                 if (i.le.0) then
                    if (i.lt.(iplot)) then
                       linbuf(i)=' '
                    else
```

```
                       linbuf(i)='x'
                       ilen=i
                    end if
c
                 end if
c
              else
c
c point is positive
                 if (i.ge.0) then
                    if (i.le.iplot) then
                       linbuf(i)='x'
                       ilen=i
                    end if
c
                 else
                    linbuf(i)=' '
                 end if
              end if
1500       continue
c
c output line
           write(idev,'(1x,i4,2x,f6.3,2x,51a1)')
     +      ipoint,stuff(ipoint),(linbuf(i2),i2=-25,ilen)
c
2000    continue
c
c
        return
        end
```