# NAG Library Function Document

# nag_tsa_auto_corr_part (g13acc)

## 1 Purpose

nag_tsa_auto_corr_part (g13acc) calculates partial autocorrelation coefficients given a set of autocorrelation coefficients. It also calculates the predictor error variance ratios for increasing order of finite lag autoregressive predictor, and the autoregressive parameters associated with the predictor of maximum order.

## 2 Specification

```
#include <nag.h>
#include <nagg13.h>
void nag_tsa_auto_corr_part (const double r[], Integer nk, Integer nl,
     double p[], double v[], double ar[], Integer *nvl, NagError *fail)
```

## 3 Description

The data consist of values of autocorrelation coefficients $r_1, r_2, \ldots, r_K$, relating to lags $1, 2, \ldots, K$. These will generally (but not necessarily) be sample values such as may be obtained from a time series $x_t$ using nag_tsa_auto_corr (g13abc).

The partial autocorrelation coefficient at lag $l$ may be identified with the parameter $p_{l,l}$ in the autoregression

$$x_t = c_l + p_{l,1}x_{t-1} + p_{l,2}x_{t-2} + \cdots + p_{l,l}x_{t-l} + e_{l,t}$$

where $e_{l,t}$ is the predictor error.

The first subscript $l$ of $p_{l,l}$ and $e_{l,t}$ emphasizes the fact that the parameters will in general alter as further terms are introduced into the equation (i.e., as $l$ is increased).

The parameters are determined from the autocorrelation coefficients by the Yule–Walker equations

$$r_i = p_{l,1}r_{i-1} + p_{l,2}r_{i-2} + \cdots + p_{l,l}r_{i-l}, \quad i = 1, 2, \ldots, l$$

taking $r_j = r_{|j|}$ when $j < 0$, and $r_0 = 1$.

The predictor error variance ratio $v_l = \text{var}\left(e_{l,t}\right)/\text{var}\left(x_t\right)$ is defined by

$$v_l = 1 - p_{l,1}r_1 - p_{l,2}r_2 - \cdots - p_{l,l}r_l.$$

The above sets of equations are solved by a recursive method (the Durbin–Levinson algorithm). The recursive cycle applied for $l = 1, 2, \ldots, (L-1)$, where $L$ is the number of partial autocorrelation coefficients required, is initialized by setting $p_{1,1} = r_1$ and $v_1 = 1 - r_1^2$.

Then

$$\begin{aligned} p_{l+1,l+1} &= \left(r_{l+1} - p_{l,1}r_l - p_{l,2}r_{l-1} - \cdots - p_{l,l}r_1\right)/v_l \\ p_{l+1,j} &= p_{l,j} - p_{l+1,l+1}p_{l,l+1-j}, \quad j = 1, 2, \ldots, l \\ v_{l+1} &= v_l\left(1 - p_{l+1,l+1}\right)\left(1 + p_{l+1,l+1}\right). \end{aligned}$$

If the condition $\left|p_{l,l}\right| \geq 1$ occurs, say when $l = l_0$, it indicates that the supplied autocorrelation coefficients do not form a positive definite sequence (see Hannan (1960)), and the recursion is not continued. The autoregressive parameters are overwritten at each recursive step, so that upon completion the only available values are $p_{Lj}$, for $j = 1, 2, \ldots, L$, or $p_{l_0-1,j}$ if the recursion has been prematurely halted.

## 4 References

Box G E P and Jenkins G M (1976) *Time Series Analysis: Forecasting and Control* (Revised Edition) Holden–Day

Durbin J (1960) The fitting of time series models *Rev. Inst. Internat. Stat.* **28** 233

Hannan E J (1960) *Time Series Analysis* Methuen

## 5 Arguments

1: **r**[**nk**] – const double *Input*

*On entry*: the autocorrelation coefficient relating to lag $k$, for $k = 1, 2, \ldots, K$.

2: **nk** – Integer *Input*

*On entry*: $K$, the number of lags. The lags range from 1 to $K$ and do not include zero.

*Constraint*: **nk** $> 0$.

3: **nl** – Integer *Input*

*On entry*: $L$, the number of partial autocorrelation coefficients required.

*Constraint*: $0 <$ **nl** $\leq$ **nk**.

4: **p**[**nl**] – double *Output*

*On exit*: **p**$[l-1]$ contains the partial autocorrelation coefficient at lag $l$, $p_{l,l}$, for $l = 1, 2, \ldots,$ **nvl**.

5: **v**[**nl**] – double *Output*

*On exit*: **v**$[l-1]$ contains the predictor error variance ratio $v_l$, for $l = 1, 2, \ldots,$ **nvl**.

6: **ar**[**nl**] – double *Output*

*On exit*: the autoregressive parameters of maximum order, i.e., $p_{Lj}$ if **fail.code** = NE_NOERROR, or $p_{l_0-1,j}$ if **fail.code** = NE_CORR_NOT_POS_DEF, for $j = 1, 2, \ldots,$ **nvl**.

7: **nvl** – Integer * *Output*

*On exit*: the number of valid values in each of **p**, **v** and **ar**. Thus in the case of premature termination at iteration $l_0$ (see Section 3), **nvl** is returned as $l_0 - 1$.

8: **fail** – NagError * *Input/Output*

The NAG error argument (see Section 3.6 in the Essential Introduction).

## 6 Error Indicators and Warnings

**NE_CORR_NOT_POS_DEF**

The autocorrelation coefficients do not form a positive definite sequence.

**NE_INT**

On entry, **nk** = ⟨*value*⟩.
Constraint: **nk** $> 0$.

On entry, **nl** = ⟨*value*⟩.
Constraint: **nl** $> 0$.

**NE_INT_2**

On entry, **nk** = ⟨*value*⟩ and **nl** = ⟨*value*⟩.
Constraint: **nk** ≥ **nl**.

## 7 Accuracy

The computations are believed to be stable.

## 8 Parallelism and Performance

Not applicable.

## 9 Further Comments

The time taken by nag_tsa_auto_corr_part (g13acc) is proportional to $(\mathbf{nvl})^2$.

## 10 Example

This example uses an input series of 10 sample autocorrelation coefficients derived from the original series of sunspot numbers generated by the nag_tsa_auto_corr (g13abc) example program. The results show five values of each of the three output arrays: partial autocorrelation coefficients, predictor error variance ratios and autoregressive parameters. All of these were valid.

### 10.1 Program Text

```
/* nag_tsa_auto_corr_part (g13acc) Example Program.
 *
 * Copyright 2014 Numerical Algorithms Group.
 *
 * Mark 2, 1991.
 * Mark 8 revised, 2004.
 */

#include <nag.h>
#include <stdio.h>
#include <nag_stdlib.h>
#include <nagg13.h>

int main(void)
{

  Integer  exit_status = 0, i, nk, nl, nvl;
  NagError fail;
  double   *ar = 0, *p = 0, *r = 0, *v = 0;

  INIT_FAIL(fail);

  printf("nag_tsa_auto_corr_part (g13acc) Example Program Results\n");
  /* Skip heading in data file */
#ifdef _WIN32
  scanf_s("%*[^\n]");
#else
  scanf("%*[^\n]");
#endif
#ifdef _WIN32
  scanf_s("%"NAG_IFMT" %"NAG_IFMT"", &nk, &nl);
#else
  scanf("%"NAG_IFMT" %"NAG_IFMT"", &nk, &nl);
#endif

  if (nl > 0 && nk > 0 && nl <= nk)
    {
      if (!(ar = NAG_ALLOC(nl, double)) ||
          !(p = NAG_ALLOC(nl, double)) ||
```

```
          !(r = NAG_ALLOC(nk, double)) ||
          !(v = NAG_ALLOC(nl, double)))
        {
          printf("Allocation failure\n");
          exit_status = -1;
          goto END;
        }
    }
  else
    {
      printf("Invalid nl or nk.\n");
      exit_status = 1;
      return exit_status;
    }
  for (i = 0; i < nk; ++i)
#ifdef _WIN32
    scanf_s("%lf", &r[i]);
#else
    scanf("%lf", &r[i]);
#endif

  /* nag_tsa_auto_corr_part (g13acc).
   * Partial autocorrelation function
   */
  nag_tsa_auto_corr_part(r, nk, nl, p, v, ar, &nvl, &fail);

  if (fail.code != NE_NOERROR)
    {
      printf("Error from nag_tsa_auto_corr_part (g13acc).\n%s\n",
              fail.message);
      exit_status = 1;
    }
  if (fail.code == NE_CORR_NOT_POS_DEF)
    printf("   Only %2"NAG_IFMT" valid sets were generated\n\n", nvl);
  if (fail.code == NE_NOERROR || fail.code == NE_CORR_NOT_POS_DEF)
    {
      printf("\n Lag  Partial    Predictor error  Autoregressive\n");
      printf("      autocorrn  variance ratio   parameter\n\n");
      for (i = 0; i < nvl; ++i)
        printf(" %2"NAG_IFMT"%9.3f%16.3f%14.3f\n", i+1, p[i], v[i],
                ar[i]);
    }
 END:
  NAG_FREE(ar);
  NAG_FREE(p);
  NAG_FREE(r);
  NAG_FREE(v);
  return exit_status;
}
```

## 10.2 Program Data

```
nag_tsa_auto_corr_part (g13acc) Example Program Data
 10  5
    0.8004    0.4355    0.0328   -0.2835   -0.4505
   -0.4242   -0.2419   -0.0550    0.3783    0.5857
```

## 10.3 Program Results

```
nag_tsa_auto_corr_part (g13acc) Example Program Results

 Lag  Partial    Predictor error  Autoregressive
      autocorrn  variance ratio   parameter

  1    0.800           0.359         1.108
  2   -0.571           0.242        -0.290
  3   -0.239           0.228        -0.193
  4   -0.049           0.228        -0.014
  5   -0.032           0.228        -0.032
```

This plot shows the partial autocorrelations for all possible lag values. Reference lines are given at $\pm z_{0.975}/\sqrt{n}$.