1969

Smoothing and Estimation Derivatives of Equispaced Data

J.J. Casaletto

John R. Rice
Purdue University, jrr@cs.purdue.edu

Report Number:
69-035
SMOOTHING AND ESTIMATING DERIVATIVES OF EQUISpaced DATA

by
J. J. Casaletto and J. R. Rice
May, 1969
CSD TR 35
ABSTRACT

This paper discusses various aspects of the smoothing and estimation of derivatives of equispaced data. The background for least squares polynomial smoothing is summarized. The various alternatives for programs in a computing center's library are discussed and a particular alternative is selected as most suitable. An algorithm named SMOOTH is given (in Fortran) which implements this alternative. SMOOTH estimates the smoothed value of the data or its first or second derivative based on specified polynomial degree and number of points to enter the smoothing. The paper concludes with a discussion of methods suitable to compute large arrays of smoothing weights. There are 3 appendices which contain, respectively, explicit formulas associated with Gram polynomials, explicit formulas for the smoothing weights and tables of initial segments of arrays for computing large tables of smoothing weights.
SMOOTHING AND ESTIMATING DERIVATIVES OF EQUISPACED DATA

1. MATHEMATICAL BACKGROUND

An algorithm for estimating the values and derivatives of equispaced, tabulated functions is presented. Let \( \{(x_j, y_j)\} \) \( j = -n, -n+1, \ldots, n \) be the data with \( h = x_{j+1} - x_j \). The algorithm is based on least squares polynomial approximation and computes smoothing weight coefficients to apply to the data.

We consider the GRAN (or Tchebycheff) polynomials \( \{P_{m,n}(x)\} \) of degree \( m \). These polynomials are orthogonal

\[
\sum_{j=-n}^{n} P_{m,n}(j) P_{k,n}(j) = 0 \quad m \neq k
\]

and are given explicitly by

\[
P_{m,n}(j) = \sum_{k=0}^{m} \frac{(-1)^k (m+k)! (n+j)! (2n-k)!}{(k!)^2 (m-k)! (n+j-k)! (2n)!}
\]

Let \( Q_{m,n}(j) \) be the best least squares approximation of \( y_j \) by a polynomial of degree \( m \) on the \( 2n+1 \) points \( i = 0, \pm 1, \ldots, \pm n \). We have

\[
Q_{m,n}(j) = \sum_{k=0}^{m} a_k P_{k,n}(j)
\]

where \( a_k = \left[ \sum_{j=-n}^{n} y_j P_{k,n}(j) \right] / S_{k,n} \) and \( S_{k,n} = \sum_{j=-n}^{n} [P_{k,n}(j)]^2 \).

Therefore, we have the smoothed estimate of the data

\[
y_j \sim Q_{m,n}(j) = \sum_{k=0}^{m} \left[ \frac{\sum_{i=-n}^{n} y_i P_{k,n}(i)}{S_{k,n}} \right] P_{k,n}(j)
\]
and, in particular, we have the smoothed value for $y_o$, given by

$$y_o \sim Q_{m,n}(0) = \sum_{k=0}^{m} \left[ \frac{1}{m} \sum_{i=-n}^{n} y_i p_{k,n}(i) \right] \frac{p_{k,n}(0)}{S_{k,n}}$$

$$= \sum_{i=-n}^{n} \left[ \frac{1}{n} \sum_{k=0}^{m} \frac{p_{k,n}(i) p_{k,n}(0)}{S_{k,n}} \right] y_i.$$

The smoothing weights are denoted by

$$A_{n,i}^m = \sum_{k=0}^{m} \frac{p_{k,n}(i) p_{k,n}(0)}{S_{k,n}}.$$

Smoothed estimates for the first and second derivatives are found by differentiating $Q_{m,n}(j)$ and evaluating it for $j = 0$. Therefore,

$$\frac{dy_o}{dx} = y'_o \sim \frac{1}{h} Q'_{m,n}(0) = \frac{1}{h} \sum_{i=-n}^{n} \sum_{k=0}^{m} \frac{p_{k,n}(i) p'_{k,n}(0)}{S_{k,n}} y_i$$

and

$$\frac{d^2y_o}{dx^2} = y''_o \sim \frac{1}{h^2} Q''_{m,n}(0) = \frac{1}{h^2} \sum_{i=-n}^{n} \sum_{k=0}^{m} \frac{p_{k,n}(i) p''_{k,n}(0)}{S_{k,n}} y_i.$$

The weights here are denoted, respectively, by $b_{n,i}^m$ and $c_{n,i}^m$. See Appendix II for $A_{n,i}^m$, $b_{n,i}^m$, and $c_{n,i}^m$ for $m = 1, 3, 5$.

2. ALTERNATIVES FOR APPLICATIONS AND COMPUTING WEIGHTS

A computing center's library should contain a routine which will, when given the arrays $X$ and $Y$, compute the smoothed values for $y_o$, $y'_o$, and $y''_o$. The major consideration for such a routine is to obtain the weights $A_{n,j}^m$, $b_{n,j}^m$, and $c_{n,j}^m$. 
One possible method is to calculate, for any \( m \) and \( n \), \( P_{m,n}(j) \) for any \( j \) and to evaluate its derivatives at zero from explicit formulae. This method has the advantages that any requested weights can be calculated and the routine is relatively short. However, the computation time would be rather large.

At the other end of the spectrum, one can limit \( m,n \) and the maximum order of the derivative; and calculate and store the necessary weights on some auxiliary storage device. The advantage here is, of course, speed. However, it is disadvantageous to restrict \( m,n \) and the order of the derivative, although these might not be, in practice, great restrictions. The major disadvantage is the number of constants which must be stored. If the maximum \( m \) is 5, maximum \( n \) is 50 (i.e. 101 points), and only \( y_0, y_1, y_2 \) are allowed, then, taking into account the facts that 

\[
A_{n,j}^{2k+1} = A_{n,j}^{2k}, \quad B_{n,j}^{2k+2} = B_{n,j}^{2k} \quad C_{n,j}^{2k+3} = C_{n,j}^{2k+2} \quad \text{for} \ k = 0, 1, \ldots
\]

and the symmetry and anti-symmetry of these constants about \( j = 0 \), it requires some 10,579 constants. This can be reduced further by observing that \( A_{n,j} \) is independent of \( j \) and that 

\[
B_{n,0}^1 = B_{n,0}^3 = B_{n,0}^5 = 0.
\]

This gives 9,150 constants and involves some additional logic. This is a large block of storage for such a routine, but not overwhelmingly large if one wants the highest possible speed.

These methods represent two extremes. The first has a large capability and small storage requirements, but is slow. The second has a limited, though practical, capability and large storage requirements, but is fast. We present an algorithm which makes a compromise on the time-storage-capability relationship; that is, using some explicit formulae, setting a practical restriction on \( m \) and computing smoothed values only for \( y_0, y_1, y_2 \).
Note in Appendix II, that the numerators of the weights are polynomials in \(n\) and \(j\), while the denominators are polynomials in \(n\); therefore, the denominators are calculated separately. The numerators are denoted by \(W_{n,j}\). The procedure is to generate integers \(W_{n,j}\), form the sum \(\sum W_{n,j} y_j\), and divide by the integral denominator and the appropriate power of \(h\). Note, further, that \(n\) is fixed for each entry into the routine and thus the \(W_{n,j}\) are polynomials in \(j\), \(j = 0, \pm 1, \pm 2, \ldots, \pm n\). This suggests the method of differences to evaluate \(W_{n,j}\). This method is more efficient for evaluating polynomials at a large number of equispaced points; whereas nested evaluation is more efficient for a smaller number of evaluations.

To make this quantitative, let \(P_n(j)\) be a polynomial of degree \(n\), to be evaluated at \(k\) equispaced points. Let the unit of work be an addition, and assume one multiplication is equivalent to \(\mu\) additions (\(\mu\) is thus a machine characteristic). Then nested evaluation requires \(nk(\mu+1)\) additions. Evaluation by differences requires \((n+1)\) starting values, the construction of a difference table, and final evaluations. If the \((n+1)\) starting values are obtained from nested evaluation, then for \(k>n+1\), \(n(n+1)(\mu+1/2)+nk\) additions are necessary. The difference table method is more efficient when \((k-n-1)\) is greater than \((n+1)/2\mu\).

The difference table method can be improved be representing \(P_n(j)\) in a point-value form; i.e. by the vector \((P_n(0), P_n(1), \ldots, P_n(n))\). Using this approach, \(nk-n(n+1)/2\) additions are necessary. Hence, this method is always more efficient than the nested form when the values of \(P_n(j)\) are required, at least, at each \(j = 0, 1, \ldots, n\).
One may use another method of representing the polynomial $P_n(j)$, the "forward difference diagonal" form with step $h$; i.e. by the vector

$$(\Delta_h^n P_n(0), \Delta_h^{n-1} P_n(0), \ldots, \Delta_h P_n(0), P_n(0)),$$

where $\Delta_h^i P_n(x)$ is the $i$-th forward difference of $P_n(x)$ with step-size $h$. This representation eliminates the construction of the difference table, $P_n(0)$ is given and successive values of $P_n(j)$ are obtained by a "rippled" (using intermediate sums as summands) addition process.

Similarly, one may use the backward difference diagonal representation with step $h$; i.e.

$$(v_h^n P_n(0), v_h^{n-1} P_n(0), \ldots, v_h P_n(0), P_n(0))$$

where $v_h^i P_n(x)$ is the $i$-th backward difference of $P_n(x)$ with step-size $h$. With this form each successive $P_n(j)$ is obtained with $n$ additions. In the calculation of the next value, the diagonal is updated so that it is the backward difference diagonal for the next $x$. Thus, the vector defining the polynomial is always changing except for the first element which, of course, is the constant difference. The advantages of this form are storage, only a one-dimensional array of $n+1$ elements is needed, and ease of coding.

Consider the example $P_4(x) = x^4 + 2x^3 + 5x^2 + 6x + 1$, the backward difference diagonal at 0 with step 1 is $B = (24, -24, 12, 2, 1)$. Now $P_4(0) = B(5) = 1$, then the simple string (a simple DO loop in Fortran),


updates the diagonal and the vector $B$ is now the backward difference diagonal of $P_4(x)$ at 1. $B$ is now $(24, 0, 12, 14, 15)$, thus $P_4(1) = B(5) = 15$. 

3. REMARKS ON THE ALGORITHM "SMOOTH"

The input parameter list, Y, X, INIT, NDERV, NDEG, NPTS, LENGTH, for the function subprogram SMOOTH is described in the initial comment cards. We use the usual convention that the zero-th derivative is the function value. The limits and checks on the arguments are also described in the comment cards.

Now, if NDERV = 0 and NDEG = 0 or 1, then A^0_n,j = A^1_n,j = 1/(2n+1); hence, W_n,j = 1. If NDERV = 1 and NDEG = 1 or 2, then B^1_n,j = B^2_n,j = 3j/(n(n+1)(2n+1)); hence W_n,j = 3j. And if NDERV = 1 with NDEG = 0, or, if NDERV = 2 with NDEG = 0 or 1, then SMOOTH is set to zero.

In the remaining cases, the W_n,j are polynomials in j, and the coefficients are polynomials in n (we call these the n-polys of j). Because of symmetry the W_n,j are only generated for j = 0, 1, ..., n and the appropriate sign attached for negative j. The parameters NDERV and NDEG point to an implicit triangular array of weights, whose columns are indexed by n (corresponding to NPTS) and whose rows are indexed by j (corresponding to the (INIT+j)-th ordinate). Therefore, NPTS specifies the particular column of this array; i.e. W^_NPTS,j for j = 0, 1, ..., NPTS. SMOOTH places no restriction on NPTS.

Let m be the greatest integer in (k+1)/2 where k is the degree of W_n,j as a polynomial in j. The procedure in SMOOTH is to evaluate W_n,j for j = 0, 1, ..., n, storing them as B(k-m+1) = W_n,j, then reflecting the appropriate values into B(1), ..., B(k-m). Thus the B-vector is now a point-value form of W_n,j. After using these m weights, the B-vector is manipulated so as to become the backward difference diagonal at j = n with step 1. Now SMOOTH continually updates the B-vector,
using the $W_{n,j} = 8(k+1)$ in the sum $\sum_j W_{n,j}(Y(INIT+j)+SIGM*Y(INIT-j))$.
where $SIGM = \pm 1$ as appropriate.

In order to evaluate the initial $W_{n,j}$, we use nested evaluation.
The $n$-polys are evaluated separately and combined with the powers of $j$.
Note that all the $n$-polys are also polynomials in $n(n+1)$.

We note that for large $n$, the $W_{n,j}$ and the denominators are very
large integers. Thus the use of integer arithmetic is limited by machine
word length. SMOOTH, as presented, uses floating point, single precision
arithmetic. If exact values for these coefficients are desired, we can
scale down the $W_{n,j}$'s and their denominators by canceling a common factor,
and/or using double precision arithmetic. The following common factors
of the numerators and denominators exist:

\[
\begin{align*}
A_{n,j}^3 &= 3 & A_{n,j}^5 &= 2^2 \cdot 3 \cdot 5 = 60 \\
B_{n,j}^3 &= 2 \cdot 3 \cdot 5 = 30 & B_{n,j}^5 &= 2^4 \cdot 3^3 \cdot 5 \cdot 7 = 15120 \\
C_{n,j}^3 &= 2 \cdot 3 \cdot 5 = 30 & C_{n,j}^5 &= 2^2 \cdot 3^3 \cdot 5 \cdot 7 = 3780 \\
\end{align*}
\]

4. THE COMPUTATION OF TABLES OF SMOOTHING WEIGHTS

SMOOTH is designed to calculate and use one specific set of weights.
To obtain a routine to produce tables of these weights the starting
values, $W_{n,0}$, ..., $W_{n,m}$ ($m$ is the greatest integer in $(k+1)/2$ where $k$
is the degree of $W_{n,j}$ as a polynomial in $j$), for each column are most
efficiently generated by difference methods. For each pair ($NDERV$, $NDEG$), $NDEG=3,5$,
one would use the point-value forms $(W_{n,j}, W_{n+1,j}, \ldots, W_{n,j})$ for
$j = 0, 1, \ldots, m$, where:

\[
(1) \quad N = \frac{NDEG+1}{2}, \text{ a minimum NPTS for } NDEG,
\]
(2) \( \phi = L^N \) where \( L \) is the degree of \( \Phi_{n,j} \) as a polynomial in \( n \), and

(3) \( n \) as above

and the point-value form for the denominator, in lieu of using any explicit formulas. This gives the initial segments of the first \((L+1)\) columns. Each column can be completed as in SMOOTH. To obtain the initial segments for additional columns, the \( m+1 \) vectors, above, should be manipulated so that they are backward difference diagonals at \( n=M \), and then updated for each column.

Two other methods for table generation are obtained by replacing the point-value forms, for the \( \Phi_{n,j} \), noted above, by either the forward or the backward difference diagonals at \( j = 0 \). The use of these diagonals facilitates program coding. Appendix III contains all the point-value, forward difference diagonal, and backward difference diagonal vectors for the cases allowed in SMOOTH. Note that the common factors have been canceled.

Finally, we note that SMOOTH may be used to compute tables of smoothing weights with its range of allowable arguments. One inserts write statements at the appropriate points (indicated by comment cards) and runs SMOOTH through the range of desired values of NDERV, NDEG and NPTS. This approach is much less efficient in computation time (and restricts the range of NDERV and NDEG), but it requires a trivial modification of SMOOTH.
FUNCTION SMOOTH(Y,X,INIT,NDERV,NPNTS,LENGTH)

C SMOOTH OPERATES ON AN EQUISpaced DATA FUNCTION (X,Y) TO PRODUCE THE
C SMOOTHED VALUE OF THE FUNCTION OR ITS 1ST OR 2ND DERIVATIVES, AT A
C SPECIFIED POINT. THE USER MUST SPECIFY THE DEGREE AND THE NUMBER OF
C POINTS, AS DESCRIBED IN THE PARAMETER DESCRIPTION BELOW TO BE USED
C IN SMOOTHING. SMOOTHING IS DONE BY APPLYING WEIGHTS TO THE NEIGHBORING
C ORIGINATES. THE WEIGHTS ARE determined BY THE LEAST SQUARES
C APPROXIMATION USING GRAM POLYNOMIALS.

C PARAMETERS
C Y *** THE ARRAY OF ORIGINATES
C X *** THE ARRAY OF X COORDINATES USED ONLY TO COMPUTE THE STEP-SIZE H.
C INIT *** THE SMOOTHED VALUE IS DESIRED AT X(INIT).
C NDERV *** THE ORDER OF THE DERIVATIVE WITH THE USUAL CONVENTION
C CONCERNING 0, THE LIMITS AND NDERV= 0, 1, 2.
C NPNTS *** THE DEGREE OF THE APPROXIMATING POLYNOMIAL.
C THE LIMITS ARE 0 TO 5
C LENGTH *** THE NUMBER OF POINTS TO BE USED IS 2NPNTS+1
C LENGTH+1 POINTS ON BOTH SIDES OF X(INIT)
C LENGTH *** THE NUMBER OF ELEMENTS IN THE ARRAY X.
C LENGTH+1 POINTS ON BOTH SIDES OF X(INIT)
C LENGTH+1 ELIMINATE THIS CHECK

DIMENSION X(1),Y(1),U(6),V(5),W(5),E(5)
REAL X2

C STEP-SIZE CALCULATION
H=1
IF(LASTP.1.E.0.1) H=X(1)
IF(LASTP.5.E.0.1) LENGTH=INIT+NPNTS
LENGTH=INIT+5
IF(LASTP.5.E.0.1) H=X(INIT+1)-X(INIT)

C PARAMETER CHECKS
DD 1 =1,5
1 MESSAGE(1)=
IF(NDERV.LT.0.OR.NDERV.GT.2) MESSAGE(1)=1
IF(NDERV.LT.0.OR.NDERV.GT.5) MESSAGE(2)=1
IF(NDERV.GT.NPNTS) MESSAGE(3)=1
IF(INIT.LE.0) MESSAGE(4)=1
IF(INIT.LE.0.OR.INIT+NPNTS.GT.LENGTH) MESSAGE(5)=1
IF(NPNTS.GT.5) MESSAGE(6)=1
GO TO 1
WRITE(6,?)
2 FORMAT(3/46H PARAMETER ERROR IN CALL TO SMOOTHING ROUTINE )
WRITE(6,?)
3 FORMAT(3/46H PARAMETER ERROR NOT EQUIPPED FOR DERIVATIVE OF ORDER 2)
WRITE(6,?)
4 FORMAT(3/46H PARAMETER ERROR NOT EQUIPPED FOR SMOOTHING BY DEGREE 2)
WRITE(6,?)
5 FORMAT(3/46H NO UNIQUE SOLUTION FOR DEGREE 2,13N BASED ON 2(13,10M
51)+1 POINTS)
6 FORMAT 4'H DERIVATIVE REQUESTED AT NEGATIVELY INDEXED POINT
   IF (I3 > NPT) RETURN
   UNITE (7)
7 FORMAT 13M ARRAY SIZE OF 13+jun INDICATES NOT ENOUGH POINTS TO ACC
   72AT NUMBER PLUS ONE POINTS ARE TO BE USED IN THE APPROXIMATION
   RETURN

C SMOOTHING

9 IF (NDEG < .DERV +.LT. 0) RETURN
L=1
SIGA=1.5
NPTS=NPTS+I
TO=1-NPTS+I
T2=1-NPTS
IF (NDEG = 1) 9, 10, 13
   RETURN

C UN DERIVED SMOOTHING

10 IF (NDEG = .LT. 1) GO TO 11
C DEGREE 0 OR 1
   SIGA = Y(1) + Y(1) - Y(INIT) + Y(INIT - 1)
   SIGA = SIGA / TO
   RETURN

11 DEGREE = (L**2+12) * TUN
   IF (NDEG = .LT. 3) GO TO 12
C DEGREE 2 OR 3
   A(1) = .9 - 6.2 - 3.6
   A(2) = .6 * (L**2) - 19.6
   GO TO 22
C DEGREE 4 OR 5
   A(3) = (L**2) + (L**2) * (L**2) + 15.6
   B(4) = (L**2) + (L**2) + 232.5
   B(5) = (L**2) + (L**2) + 232.5
   SIGA = A + B * TUN
   SIGA = SIGA / TO
   GO TO 22

C 2ND DERIVATIVE SMOOTHING

12 DEGREE = (L**2+12) * NC + 2 * TO
   NC = 2 * TO
   IF (NDEG = .LT. 3) GO TO 13
C DEGREE 2 OR 3
   C(1) = .9 - 6.2 - 3.6
   C(2) = .6 * (L**2) - 19.6
   GO TO 22
C DEGREE 4 OR 5
   C(3) = (L**2) + (L**2) * (L**2) + 15.6
   D(4) = (L**2) + (L**2) + 232.5
   D(5) = (L**2) + (L**2) + 232.5
   SIGA = A + B * TO
   SIGA = SIGA / TO
   GO TO 22
B(4)=U(3)+1660V.\#N2-27U090.\#N2+1690U.
B(5)=U(3)+13020V.\#N2-21760.\#N2+1701U0U.
GO TO 20

C 1ST DERIVATIVE SMOOTHING

15 DENOM=\#N2+TDH
SIGN=1
IF(IND<=LT.2) GO TO 17
C DEGRE 1 OR 2
DO 16 K=2,NPTS
16 SMOOTH=SIGN*FLOAT(K)\*(Y(INIT+K)-Y(INIT-K))
SMOOTH=3.\%SMOOTH/DENOM
RETURN

17 DENOM=\#E,0.*(T2+12.*\#N2-2.)
IF(\#DEMOU(T,4)) GO TO 16
C DEGRE 3 OR 4
B(2)=U
B(3)=75.\#N2-160.\#N2+60.
B(4)=150.\#N2-90.\#N2+30U.
GO TO 21
C DEGRE 5
19 B(5)=4.*T2*(\#N2-6.1)\#LEN0.:
B(3)=U
C A(1),A(2) AND A(6) ARE USED HERE AS TEMP STORAGE
A(1)=14.4*(R-2-Y1)\#N2+90.
A(2)=15.4*(R-50-Y2)*N2+12.
B=U(A(1))*A(12).\#N2-1400.
B(1)=U(A(3))\#N2-300.\#N2+10U*+11\#U(2)*\#U(2).
B(6)=-77.4\#N2+1155.
DO 19 J=-6
C AS STEP-SIZE ALREADY INCLUDED IN DEGREE USED HERE AS TEMP STORAGE.
H=J-3
19 B(J)\#U(Y3)\#U(R+J(6))\#U(2)+U(2)\#U(2)\#U(1)*"*

C SET U-VECTOR TO P-V FORM AND GENERATE PARTIAL SUMS.
L=2
20 KEY=3
21 LAL=1
22 IF(KEY=1)SIGN=U(KEY+1)
IF(KEY=.LT.3)SIGN=U(3)
SMOOTH=SIGN*Y(INIT)
23 DO 24 K=1,L
C TO QUT IN THE FIRST PART OF U(K,J) ARRAY, PRINT U(K,J) HERE.
24 SMOOTH=SMOOTH+U(KEY+K)*Y(INIT+K)+SIGN*Y(INIT-K)
IP <-> (KEY+L) GO TO 26

C TRANSFORM U-VECTOR INTO CHUNKY DIffERENCE DIAGONAL
\#KEY+L
\#L=1
DO 25 I=1,4
   K=K+1
DO 25 J=1,K
25 B(J)=U(J+1)-U(J)

C GENERATE AND STORE WEIGHTS INTO SUM
   M=1
DO 27 K=1,M
   MPTS
DO 26 J=1,M
26 B(J)=B(J)+B(J-1)

C TO OBTAIN THE REST OF THE Y(I,J) ARRAY, PRINT 011 HERE.
27 SMOOTH=SMOTH+U(J)*(Y(INIT+K)+SIG.*Y(INIT-K))
C UCRON INCLUDES STEP-SIZE ADJUSTMENT
26 SMOOTH=SMOOTH/UCRON
RETURN
END
APPENDIX I: GRAM POLYNOMIALS, DERIVATIVES AND SPECIAL VALUES

A. THE GRAM POLYNOMIALS

\[ P_{0,n}(j) = 1 \]
\[ P_{1,n}(j) = \frac{1}{n} \]
\[ P_{2,n}(j) = \frac{3j^2-n(n+1)}{n(2n-1)} \]
\[ P_{3,n}(j) = \frac{5j^3-(3n^2+3n-1)j}{n(n-1)(2n-1)} \]
\[ P_{4,n}(j) = \frac{35j^4-5(6n^2+6n-5)j^2+3n(n^2-1)(n+2)}{2n(n-1)(2n-1)(2n-3)} \]
\[ P_{5,n}(j) = \frac{63j^5-35(2n^2+2n-3)j^3+(15n^4+30n^3-35n^2-50n+12)j}{2n(n-1)(n-2)(2n-1)(2n-3)} \]

B. THE DERIVED POLYNOMIALS

\[ P'_{0,n}(j) = 0 \]
\[ P'_{1,n}(j) = \frac{1}{n} \]
\[ P'_{2,n}(j) = \frac{6j}{n(2n-1)} \]
\[ P'_{3,n}(j) = \frac{15j^2-(3n^2+3n-1)}{n(2n-1)(n-1)} \]
\[ P'_{4,n}(j) = \frac{10j(14j^2-(6n^2+6n-5))}{2n(n-1)(2n-1)(2n-3)} \]
\[ P'_{5,n}(j) = \frac{315j^4-105(2n^2+2n-3)j^2+(15n^4+30n^3-35n^2-50n+12)}{2n(n-1)(n-2)(2n-1)(2n-3)} \]

C. THE SECOND DERIVATIVES

\[ P''_{0,n}(j) = 0 \]
\[ P''_{1,n}(j) = 0 \]
\[ P''_{2,n}(j) = \frac{6}{n(2n-1)} \]
\[ P''_{3,n}(j) = \frac{30j}{n(5-n)(2n-1)} \]
\[ P''_{4,n}(j) = \frac{420j^2-10(6n^2+6n-5)}{2n(n-1)(2n-1)(2n-3)} \]
\[ P''_{5,n}(j) = \frac{1260j^3-210(2n^2+2n-3)j}{2n(n-1)(n-2)(2n-1)(2n-3)} \]
D. SPECIAL VALUES

\[ p_{k,n}(0) = 0, \text{ for } k \text{ odd} \]

\[ p_{0,n}(0) = 1 \]

\[ p_{2,n}(0) = \frac{n+1}{2n-1} \]

\[ p_{4,n}(0) = \frac{3(n+1)(n+2)}{2(2n-1)(2n-3)} \]

\[ p'_{k,n}(0) = 0, \text{ for } k \text{ even} \]

\[ p'_{1,n}(0) = \frac{1}{n} \]

\[ p'_{3,n}(0) = \frac{5n^2+3n-1}{n(n-1)(2n-1)} \]

\[ p'_{5,n}(0) = \frac{25n^4+30n^3-50n^2-50n+12}{2n(2n-1)(n-2)(2n-1)(2n-3)} \]

\[ p''_{k,n}(0) = 0, \text{ for } k=0 \text{ and } k \text{ odd} \]

\[ p''_{2,n}(0) = \frac{6}{n(2n-1)} \]

\[ p''_{4,n}(0) = \frac{5(6n^2+6n+5)}{n(n-1)(2n-1)(2n-3)} \]

E. THE SUM OF THE SQUARES

\[ S_{0,n} = 2n+1 \]

\[ S_{1,n} = \frac{(n+1)(2n+1)}{3n} \]

\[ S_{2,n} = \frac{(n+1)(2n+1)(2n+3)}{5n(2n-1)} \]

\[ S_{3,n} = \frac{(n+1)(n+2)(2n+1)(2n+3)}{7n(n-1)(2n-1)} \]

\[ S_{4,n} = \frac{(n+1)(n+2)(2n+1)(2n+3)(2n+5)}{9n(n-1)(2n-1)(2n-3)} \]

\[ S_{5,n} = \frac{(n+1)(n+2)(n+3)(2n+1)(2n+3)(2n+5)}{11n(n-1)(n-2)(2n-1)(2n-3)} \]
APPENDIX II: FORMULAE FOR THE WEIGHTS

NOTATION:

\[ a_{n,j}^m = \sum_{k=0}^{m} \frac{p_{k,n}(j)}{S_{k,n}} \]
\[ b_{n,j}^m = \sum_{k=0}^{m} \frac{p_{k,n}(j) p_{k,n}'(0)}{S_{k,n}} \]
\[ c_{n,j}^m = \sum_{k=0}^{m} \frac{p_{k,n}(j) p_{k,n}''(0)}{S_{k,n}} \]

denotes the j-th weight for m-th degree smoothing based on 2n+1 points.

denotes the j-th weight for the 1st derivative by m-th degree smoothing based on 2n+1 points.

denotes the j-th weight for the 2nd derivative by m-th degree smoothing based on 2n+1 points.

\[ A_{n,j}^1 = \frac{1}{2n+1} \]
\[ A_{n,j}^3 = \frac{3[(3n^2+3n-1)-5j^2]}{(2n-1)(2n+1)(2n+3)} \]
\[ A_{n,j}^5 = \frac{15(63n^4-35(2n^2+2n-3)j^2+(15n^4+30n^2-35n^2-50n+12))(2n^2-7n+6)}{4(2n-1)(2n+1)(2n+3)(2n+5)} \]

\[ b_{n,j}^1 = \frac{3j}{n(n+1)(2n+1)} \]
\[ b_{n,j}^3 = \frac{25[3n^4+6n^3-3n^2-3n+1]j-35[3n^2+3n-1]}{n(n+1)(n+2)(2n+1)(2n+3)(2n+5)} \]
\[ b_{n,j}^5 = \frac{(693[15n^4+30n^3-35n^2-50n+12]j^5-35[4(3n^2+3n-1)(2n^2+11n+15)(2n^2-2n+2)+12(2n^2+11n+15)+11(15n^4+30n^3-35n^2-50n+12)]j^2} {4n(n-2)(n+1)(n+2)(n+3)(2n-3)(2n-1)(2n+1)(2n+3)(2n+5)} \]
\[ c_{n,j}^1 = 0 \]

\[ c_{n,j}^3 = \frac{30[3j^2-n(n+1)]}{n(n+1)(2n-1)(2n+1)(2n+3)} \]

\[ c_{n,j}^5 = \frac{-15\{105(6n^2+6n-5)\}^4 - 3(196n^4+392n^3-196n^2-392n+245)j^2 + (70n^6+210n^5-35n^4 - 420n^3+35n^2+210n)}{2n(n-1)(n+1)(n+2)(2n-3)(2n-1)(2n+1)(2n+3)(2n+5)} \]
NESTED FORM FOR NUMERATORS

\[ a_{n,j}^1 = 1 \]
\[ a_{n,j}^2 = (9 \cdot n + 9) \cdot n - 3 - 15 \cdot j \cdot j \]
\[ a_{n,j}^3 = [945 \cdot j \cdot j - (1050n+1050)n-1575)] \cdot j \cdot j + ((15n+30)n-35)n-50)n+12 \]
\[ b_{n,j}^1 = 5 \cdot j \]
\[ b_{n,j}^2 = [((-105n-105)n+35) \cdot j \cdot j + ((75n+150)n-75)n+25] \cdot j \]
\[ b_{n,j}^3 = \{ (((1035n+20790)n-24255)n-34650)n+8316)] \cdot j \cdot j + (((-13230n-39690)n+33075)n+132300)n-37485)n-110250)n+26460] \cdot j \cdot j \]
\[ + (((((3675n+14700)n-7350)n-73500)n-13965)n+111720)n+26460)n-44100)n+10584) \cdot j \]
\[ c_{n,j}^1 = 0 \]
\[ c_{n,j}^2 = 90 \cdot j \cdot j - (30n+30)n \]
\[ c_{n,j}^3 = [((-9450n-9450)n+7875) \cdot j \cdot j + (((8820n+17640)n-8820)n-17640)n+11025) \cdot j \cdot j \]
\[ ((((-1050n-3150)n+525)n-6300)n+525)n-3150)n \]
SPECIAL FORMS FOR NUMERATORS, \( W_{n,o} \ldots, W_{n,m} \)

LET \( \delta = (n+1)n \)

\[ \begin{align*}
\Lambda^1_{n,j} & : 1 \\
\Lambda^3_{n,o} & : 96-3 \\
\Lambda^3_{n,1} & : A^3_{n,0} - 15 \quad \text{(i.e. numerator of } A^3_{n,0}) - 15 \\
\Lambda^5_{n,0} & : (2256-750)\delta + 180 \\
\Lambda^5_{n,1} & : A^5_{n,0} - 10506 + 2520 \\
\Lambda^5_{n,2} & : A^5_{n,1} - 42006 + 21420 \\
B^1_{n,j} & : 3j \\
B^3_{n,o} & : 0 \\
B^3_{n,1} & : (756-180)\delta + 60 \\
B^3_{n,0} & : (1506-990)\delta + 330 \\
B^5_{n,o} & : 0
\end{align*} \]

SET: \( Z_1 = (16-30)\delta + 90 \quad Z_2 = (156-50)\delta + 12 \)

\( :l = -7706 + 1155 \quad :z = -Z_1((206-140) \quad Z_1 = Z_1((506-300)\delta + 100) + 112Z_2Z_2 \)

THEN FOR \( j = 1, 2, 3 \)

\[ B^5_{n,j} : (((693jj+n)Z_2Z_2jj+Z_1)j) \]
\begin{align*}
  c_{n,j}^1 & : 0 \\
  c_{n,0}^3 & : -306 \\
  c_{n,1}^3 & : c_{n,0}^3 + 90 \\
  c_{n,0}^5 & : ((-10506 + 3675) \times 3 - 3150) \times 6 \\
  c_{n,1}^5 & : c_{n,0}^5 + (8820 - 27050) \times 4 + 18000 \\
  c_{n,2}^5 & : c_{n,0}^5 + (35280 - 221760) \times 6 + 170100
\end{align*}
APPENDIX III. Initial Segments of Special Arrays for Computing Weights by Differences

Values are given in the following tables so that complete tables of weights may be made by differences without any direct evaluation. Also given are the factors which will appear in both numerator and denominator of the formulae. These factors have been cancelled in these tables.

**TABLE I: 3RD DEGREE SMOOTHING**

<table>
<thead>
<tr>
<th>( \lambda^n )</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>( A_{n,j}^3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>17</td>
<td>35</td>
<td>59</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>12</td>
<td>30</td>
<td>54</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\( S_{3,n} = \text{denom} \)

factors 3

\( A_{n,j}^3 \)

even in \( j \), degree 2

**TABLE II: 5TH DEGREE SMOOTHING**

<table>
<thead>
<tr>
<th>( \lambda^n )</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>( A_{n,j}^5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>393</td>
<td>1253</td>
<td>3003</td>
<td>6093</td>
<td>11063</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>225</td>
<td>945</td>
<td>2520</td>
<td>5400</td>
<td>10125</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>-90</td>
<td>210</td>
<td>1260</td>
<td>3510</td>
<td>7500</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\( S_{5,n} = \text{denom} \)

factors \( 2^2 \cdot 3 \cdot 5 \)

even in \( j \), degree 4
### TABLE III: 1ST DERIVATIVE BY 3RD DEGREE SMOOTHING

<table>
<thead>
<tr>
<th>j</th>
<th>n</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>56</td>
<td>290</td>
<td>882</td>
<td>2072</td>
<td>4160</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>-7</td>
<td>335</td>
<td>1351</td>
<td>3521</td>
<td>7445</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ S_{3,n} = \text{denom} \]

factors 2 · 3 · 5
odd in j, degree 3

### TABLE IV: 1ST DERIVATIVE BY 5TH DEGREE SMOOTHING

<table>
<thead>
<tr>
<th>j</th>
<th>n</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>1485</td>
<td>20153</td>
<td>129528</td>
<td>564840</td>
<td>1926375</td>
<td>5531295</td>
<td>13972728</td>
<td>31954728</td>
<td>67481505</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>-297</td>
<td>15883</td>
<td>158508</td>
<td>823338</td>
<td>3079725</td>
<td>9351573</td>
<td>24509058</td>
<td>57518308</td>
<td>123735843</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>33</td>
<td>-9667</td>
<td>52458</td>
<td>603198</td>
<td>2930411</td>
<td>10157763</td>
<td>28816158</td>
<td>71227688</td>
<td>158937093</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ S_{5,n} = \text{denom} \]

factors 2 · 3 · 5 · 7
odd in j, degree 5
**TABLE V: 2ND DERIVATIVE BY 3RD DEGREE SMOOTHING**

<table>
<thead>
<tr>
<th>j^n</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-6</td>
<td>-12</td>
<td>-20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>-3</td>
<td>-9</td>
<td>-17</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$$S_{3,n} = \text{denom factors } 2 \cdot 3 \cdot 5$$
even in j, degree 2

**TABLE VI: 2ND DERIVATIVE BY 5TH DEGREE SMOOTHING**

<table>
<thead>
<tr>
<th>j^n</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-350</td>
<td>-1850</td>
<td>-6650</td>
<td>-18900</td>
<td>-45780</td>
<td>-98700</td>
<td>-194700</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>-95</td>
<td>-1055</td>
<td>-4760</td>
<td>-15080</td>
<td>-38859</td>
<td>-87115</td>
<td>-176440</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>335</td>
<td>755</td>
<td>35</td>
<td>-4855</td>
<td>-19751</td>
<td>-54495</td>
<td>-124355</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$$S_{5,n} = \text{denom factors } 2^2 \cdot 3^3 \cdot 5 \cdot 7$$
even in j, degree 4
Tables of Forward Differences Diagonals with Respect to $n$.

NOTE: These numbers are used right to left in the program description.

Table 1a: 3rd Degree Smoothing

<table>
<thead>
<tr>
<th>j</th>
<th>$AT_{n=2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>17 18 6</td>
</tr>
<tr>
<td>1</td>
<td>12 18 6</td>
</tr>
</tbody>
</table>

Table 2a: 5th Degree Smoothing

<table>
<thead>
<tr>
<th>j</th>
<th>$AT_{n=3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>593 860 890 450 90</td>
</tr>
<tr>
<td>1</td>
<td>225 720 855 450 90</td>
</tr>
<tr>
<td>2</td>
<td>-90 300 750 450 90</td>
</tr>
</tbody>
</table>

Table 3a: 1st Derivative by 3rd Degree Smoothing

<table>
<thead>
<tr>
<th>j</th>
<th>$AT_{n=2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0 0 0 0 0</td>
</tr>
<tr>
<td>1</td>
<td>56 234 358 240 60</td>
</tr>
<tr>
<td>2</td>
<td>-7 342 674 480 120</td>
</tr>
</tbody>
</table>

Table 4a: 1st Derivative by 5th Degree Smoothing

<table>
<thead>
<tr>
<th>j</th>
<th>$AT_{n=3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0 0 0 0 0</td>
</tr>
<tr>
<td>1</td>
<td>1485 18668 90707 235230 365056 351820 207270 60600 9800</td>
</tr>
<tr>
<td>2</td>
<td>-297 16180 126445 395760 673592 680960 410760 137200 19600</td>
</tr>
<tr>
<td>3</td>
<td>33 -9700 71825 416790 871068 964740 606690 205800 29400</td>
</tr>
</tbody>
</table>

Table 5a: 2nd Derivative by 3rd Degree Smoothing

<table>
<thead>
<tr>
<th>j</th>
<th>$AT_{n=2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-6 -6 -2</td>
</tr>
<tr>
<td>1</td>
<td>-5 -6 -2</td>
</tr>
</tbody>
</table>

Table 6a: 2nd Derivative by 5th Degree Smoothing

<table>
<thead>
<tr>
<th>j</th>
<th>$AT_{n=3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-350 -1500 -3300 -4150 -3030 -1200 -200</td>
</tr>
<tr>
<td>1</td>
<td>-95 -960 -2745 -3870 -2974 -1200 -200</td>
</tr>
<tr>
<td>2</td>
<td>335 420 -1140 -3030 -2806 -1200 -200</td>
</tr>
</tbody>
</table>
Tables of Backward Difference Diagonals with Respect to \( n \).

NOTE: These numbers are used right to left in the program description.

### Table 1a: 3rd Degree Smoothing

\[ j \quad \text{AT \( n = 2 \)} \]

<table>
<thead>
<tr>
<th>( j )</th>
<th>0</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>17</td>
<td>12</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>12</td>
<td>12</td>
<td>6</td>
</tr>
</tbody>
</table>

### Table 2b: 5th Degree Smoothing

\[ j \quad \text{AT \( n = 3 \)} \]

<table>
<thead>
<tr>
<th>( j )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>393</td>
<td>330</td>
<td>260</td>
<td>180</td>
<td>90</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>225</td>
<td>225</td>
<td>225</td>
<td>180</td>
<td>90</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>-90</td>
<td>-90</td>
<td>120</td>
<td>180</td>
<td>90</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 3a: 1st Derivative by 3rd Degree Smoothing

\[ j \quad \text{AT \( n = 2 \)} \]

<table>
<thead>
<tr>
<th>( j )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>56</td>
<td>56</td>
<td>58</td>
<td>60</td>
<td>60</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>-7</td>
<td>28</td>
<td>74</td>
<td>120</td>
<td>120</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 4b: 1st Derivative by 5th Degree Smoothing

\[ j \quad \text{AT \( n = 3 \)} \]

<table>
<thead>
<tr>
<th>( j )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1485</td>
<td>1485</td>
<td>1485</td>
<td>1482</td>
<td>1476</td>
<td>1470</td>
<td>1470</td>
<td>9800</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>-297</td>
<td>-297</td>
<td>-339</td>
<td>-456</td>
<td>-648</td>
<td>-840</td>
<td>-840</td>
<td>19600</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>33</td>
<td>-1353</td>
<td>-3061</td>
<td>-5274</td>
<td>-7992</td>
<td>-10710</td>
<td>-10710</td>
<td>29400</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 5b: 2nd Derivative by 3rd Degree Smoothing

\[ j \quad \text{AT \( n = 2 \)} \]

<table>
<thead>
<tr>
<th>( j )</th>
<th>0</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-6</td>
<td>-4</td>
<td>-2</td>
</tr>
<tr>
<td>1</td>
<td>-3</td>
<td>-4</td>
<td>-2</td>
</tr>
</tbody>
</table>

### Table 6b: 2nd Derivative by 5th Degree Smoothing

\[ j \quad \text{AT \( n = 3 \)} \]

<table>
<thead>
<tr>
<th>( j )</th>
<th>0</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-350</td>
<td>-320</td>
<td>-290</td>
</tr>
<tr>
<td>1</td>
<td>-95</td>
<td>-111</td>
<td>-127</td>
</tr>
<tr>
<td>2</td>
<td>335</td>
<td>336</td>
<td>302</td>
</tr>
</tbody>
</table>
Complete Initial Difference Tables For The Weights, Displaying The Point-Value, Forward Diagonal, and Backward Diagonal Forms At Minimum $n$.

\[ A_{3}^{1} n, j \]
\[ j = 0 \]
\[ \begin{array}{ccc}
  x & 35 & 50 \\
 12 & 18 & 24
\end{array} \]

\[ \text{point-value} \]
\[ \begin{array}{c}
  \text{backward} \\
 6 \\
 6 \\
\end{array} \]
\[ \text{forward} \]
\[ j = 1 \]
\[ \begin{array}{ccc}
  12 & 30 & 54 \\
  12 & 18 & 24 \\
  6 & 6
\end{array} \]

\[ A_{5}^{1} n, j \]
\[ j = 0 \]
\[ \begin{array}{cccccc}
  x & 393 & 1253 & 3003 & 6093 & 11063 \\
  350 & 860 & 1750 & 3020 & 4970 \\
  260 & 890 & 1540 & 1880 \\
  180 & 450 & 540 \\
  90 & 90
\end{array} \]

\[ j = 1 \]
\[ \begin{array}{cccccc}
  x & 225 & 945 & 2520 & 5400 & 10125 \\
  225 & 720 & 1575 & 2880 & 4725 \\
  225 & 855 & 1305 & 1845 \\
  180 & 450 & 540 \\
  90 & 90
\end{array} \]

\[ j = 2 \]
\[ \begin{array}{cccccc}
  x & -90 & 210 & 1260 & 3510 & 7500 \\
  -90 & 300 & 1050 & 2250 & 3990 \\
  120 & 750 & 1200 & 1740 \\
  180 & 450 & 540 \\
  90 & 90
\end{array} \]

\[ B_{3}^{1} n, j \]
\[ j = 1 \]
\[ \begin{array}{cccccc}
  x & 56 & 290 & 882 & 2072 & 4160 \\
  56 & 234 & 592 & 1190 & 2088 \\
  58 & 358 & 598 & 898 \\
  60 & 240 & 300 \\
  60 & 60
\end{array} \]

\[ j = 2 \]
\[ \begin{array}{cccccc}
  x & -7 & 335 & 1351 & 3521 & 7445 \\
  28 & 342 & 1016 & 2170 & 3924 \\
  74 & 674 & 1154 & 1754 \\
  120 & 480 & 600 \\
  120 & 120
\end{array} \]
\[ a_{n,j} \]

\[
\begin{array}{cccccccccccc}
\text{j=1} & 1485 & 20153 & 129528 & 564840 & 1926375 & 5531295 & 13972728 & 31954728 & 67481505 \\
1485 & 18668 & 109375 & 435312 & 1361535 & 5604920 & 8441433 & 17982000 & 35526777 \\
1485 & 90707 & 325937 & 926223 & 2243385 & 4836513 & 9540567 & 17544777 \\
-1482 & 235230 & 600285 & 1517162 & 2593128 & 4704054 & 8004210 \\
-1476 & 365056 & 716876 & 1275966 & 2110926 & 5300156 \\
-1470 & 351820 & 559030 & 834960 & 1189230 \\
-1470 & 207270 & 275870 & 354270 \\
-1470 & 68600 & 78400 \\
9800 & 5800 \\
\hline
\text{j=2} & 297 & 15883 & 158508 & 823338 & 3079725 & 9351575 & 24509058 & 57508308 & 123735843 \\
-297 & 16180 & 142625 & 664830 & 2256387 & 6271848 & 15157485 & 32999250 & 66227535 \\
-339 & 126445 & 522205 & 1591557 & 4015661 & 8865637 & 17841765 & 35228285 \\
-456 & 395760 & 1069352 & 2423904 & 4870176 & 8956128 & 15386520 \\
-648 & 67592 & 1354552 & 2446272 & 4085952 & 6430392 \\
-840 & 680960 & 1091720 & 1639680 & 2344440 \\
-840 & 410760 & 547960 & 704760 \\
0 & 137200 & 156800 \\
19600 & 19600 \\
\hline
\text{j=3} & 33 & -9667 & 52458 & 603198 & 2930411 & 19157763 & 28816158 & 71227658 & 158937093 \\
-1353 & -9700 & 62125 & 550740 & 2327213 & 7227352 & 18658395 & 42411500 & 87700945 \\
-3061 & 71825 & 488615 & 1776475 & 4900139 & 11431043 & 23753105 & 45297935 \\
-5274 & 416790 & 1287858 & 3123666 & 6550964 & 12320662 & 21544830 \\
-7992 & 873068 & 1835808 & 3407238 & 5791158 & 9222768 \\
-10710 & 964740 & 1571430 & 2383920 & 3431610 \\
-10710 & 606690 & 812490 & 1047690 \\
0 & 205800 & 235200 \\
29400 & 29400 \\
\end{array}
\]
\[ c^3_{n,j} \]

<table>
<thead>
<tr>
<th></th>
<th>-6</th>
<th>-12</th>
<th>-20</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-4</td>
<td>-6</td>
<td>-8</td>
</tr>
<tr>
<td>1</td>
<td>-2</td>
<td>-2</td>
<td></td>
</tr>
</tbody>
</table>

\[ c^5_{n,j} \]

<table>
<thead>
<tr>
<th></th>
<th>-350</th>
<th>-1350</th>
<th>-6650</th>
<th>-18900</th>
<th>-45780</th>
<th>-98700</th>
<th>-194700</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-320</td>
<td>-1500</td>
<td>-4800</td>
<td>-12250</td>
<td>-26880</td>
<td>-52920</td>
<td>-96000</td>
</tr>
<tr>
<td>1</td>
<td>-290</td>
<td>-3300</td>
<td>-7450</td>
<td>-14630</td>
<td>-26040</td>
<td>-43080</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>-260</td>
<td>-4150</td>
<td>-7180</td>
<td>-11410</td>
<td>-17040</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>-230</td>
<td>-3030</td>
<td>-4230</td>
<td>-5630</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>-200</td>
<td>-1200</td>
<td>-1400</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>-200</td>
<td>-200</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>-95</th>
<th>-1055</th>
<th>-4760</th>
<th>-15080</th>
<th>-38859</th>
<th>-87115</th>
<th>-176440</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-111</td>
<td>-960</td>
<td>-3705</td>
<td>-10320</td>
<td>-23779</td>
<td>-48256</td>
<td>-89325</td>
</tr>
<tr>
<td>1</td>
<td>-127</td>
<td>-2745</td>
<td>-6615</td>
<td>-13459</td>
<td>-24477</td>
<td>-41069</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>-148</td>
<td>-3870</td>
<td>-6844</td>
<td>-11018</td>
<td>-16592</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>-174</td>
<td>-2974</td>
<td>-4174</td>
<td>-5574</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>-200</td>
<td>-1200</td>
<td>-1400</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>-200</td>
<td>-200</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ c^7_{n,j} \]

<table>
<thead>
<tr>
<th></th>
<th>-335</th>
<th>755</th>
<th>35</th>
<th>-4855</th>
<th>-19751</th>
<th>-54495</th>
<th>-124335</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-336</td>
<td>420</td>
<td>-720</td>
<td>-4800</td>
<td>-14896</td>
<td>-34744</td>
<td>-69840</td>
</tr>
<tr>
<td>1</td>
<td>302</td>
<td>-1140</td>
<td>-4170</td>
<td>-10006</td>
<td>-19848</td>
<td>-35096</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>188</td>
<td>-3030</td>
<td>-5836</td>
<td>-9842</td>
<td>-15248</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>-6</td>
<td>-2806</td>
<td>-4006</td>
<td>-5406</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>-200</td>
<td>-1200</td>
<td>-1400</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>-200</td>
<td>-200</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>