DTIC FILE COPY

NSWC TR 87-372

AD-A199 341

A METHOD FOR THE CALCULATION OF ABSCISSAS AND WEIGHT FACTORS USING GAUSSIAN INTEGRATION FOR INTEGRANDS WITH A LOGARITHMIC SINGULARITY

BY   STEPHEN A. WILKERSON

RESEARCH AND TECHNOLOGY DEPARTMENT

20 NOVEMBER 1987

DTIC
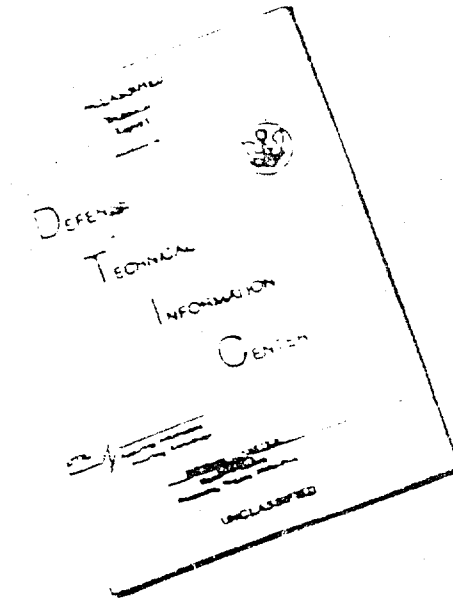ELECTE
SEP 1 4 1988
E

NAVAL SURFACE WARFARE CENTER

Dahlgren, Virginia 22448-5000 ● Silver Spring, Maryland 20903-5000

88 9 14 147

DISCLAIMER NOTICE

THIS DOCUMENT IS BEST
QUALITY AVAILABLE. THE COPY
FURNISHED TO DTIC CONTAINED
A SIGNIFICANT NUMBER OF
PAGES WHICH DO NOT
REPRODUCE LEGIBLY.

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b RESTRICTIVE MARKINGS		
2a. SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.		
2b DECLASSIFICATION/DOWNGRADING SCHEDULE			
4. PERFORMING ORGANIZATION REPORT NUMBER(S) NSWC TR 87-372	5 MONITORING ORGANIZATION REPORT NUMBER(S)		

6a NAME OF PERFORMING ORGANIZATION Naval Surface Warfare Center	6b OFFICE SYMBOL (If applicable) R14	7a NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) White Oak Laboratory 10901 New Hampshire Avenue Silver Spring, MD 20903-5000		7b ADDRESS (City, State, and ZIP Code)

8a. NAME OF FUNDING/SPONSORING ORGANIZATION	8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER		
8c. ADDRESS (City, State, and ZIP Code)		10 SOURCE OF FUNDING NUMBERS		
		PROGRAM ELEMENT NO	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
A Method for the Calculation of Abscissas and Weight Factors Using
Gaussian Integrands with a Logarithmic Singularity

12. PERSONAL AUTHOR(S)
Wilkerson, Stephen A.

13a. TYPE OF REPORT Final	13b TIME COVERED FROM 6/87 TO 9/87	14 DATE OF REPORT (Year, Month, Day) 1987, November, 20	15 PAGE COUNT 32

16. SUPPLEMENTARY NOTATION

17.	COSATI CODES		18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Abscissas; Gaussian integration;
12	02		Orthogonal polynomials.

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

A method for the calculation of abscissas and weight factors using
Gaussian integration for integrands with a logarithmic singularity is
presented. The method shows good convergent properties and allows for
the accurate estimation of the error. A program is supplied for the
generation of orthogonal polynomials with weight Log(x) to order n, and
numerical tables for the Gaussian integration method are provided.

20 DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a NAME OF RESPONSIBLE INDIVIDUAL Stephen A. Wilkerson	22b TELEPHONE (Include Area Code) (202) 394-3971	22c OFFICE SYMBOL R14

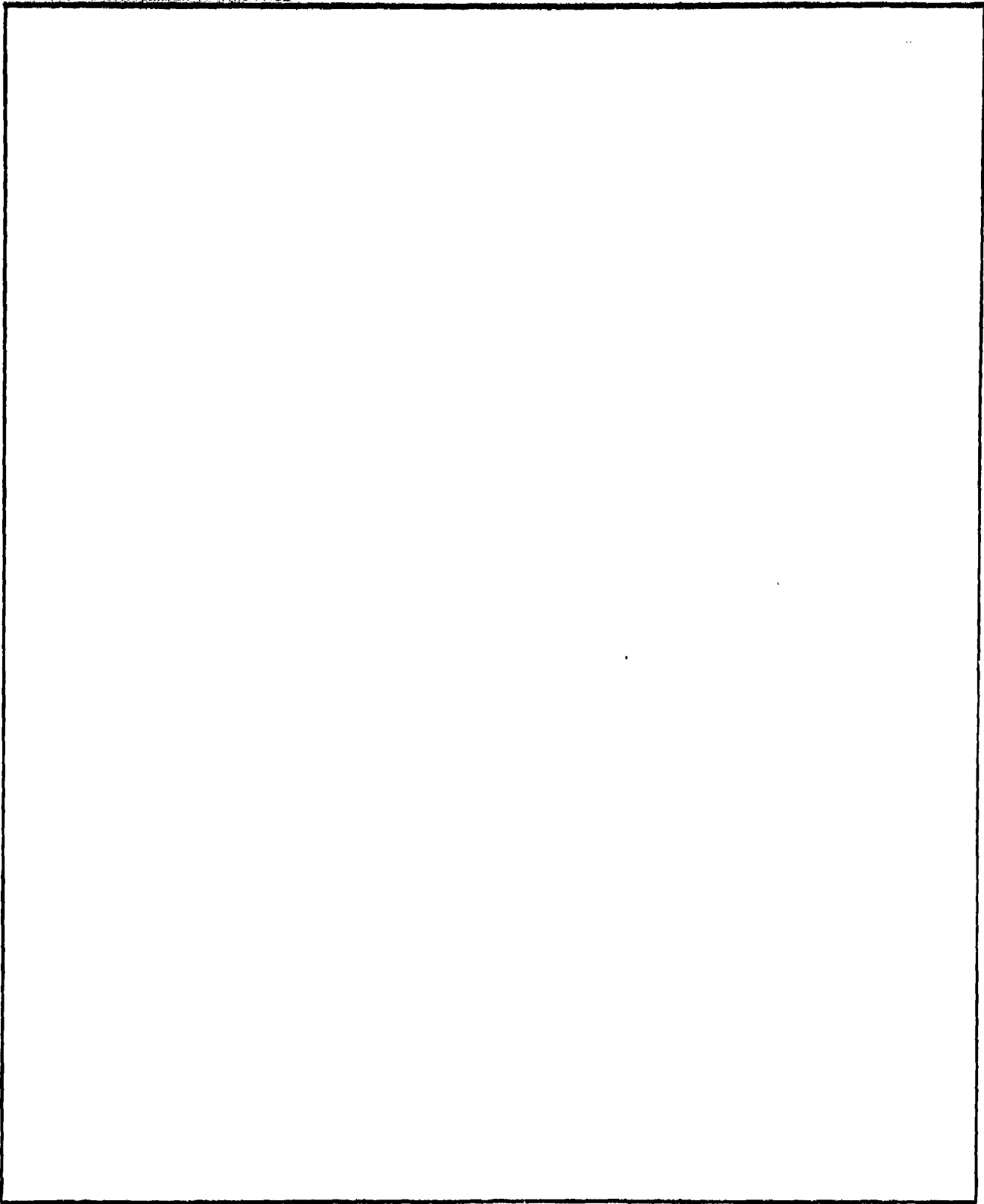**DD FORM 1473,** 84 MAR — 83 APR edition may be used until exhausted
All other editions are obsolete

FOREWORD

This work was sponsored under the auspices of the Naval Surface Warfare Center's long term study program. This program allows employees the opportunity of continued academic study for the period of 1 year. This study was conducted during the summer of 1987 under the aforementioned program. The purpose of this study was to approximate logarithmic singularities found in integrals by use of a Gaussian integration formulation. The method provides a simple approach to the calculation of Gaussian integration weight factors and roots. A short program is also supplied to future users on the method for similar singularities which occur in physical problems.

The author gratefully acknowledges the advice and suggestions of Dr. A. Prosperetti and Dr. O. Hassan of the Department of Mechanics, the Johns Hopkins University, who provided the technical foundation for this work.

Approved by:

K. F. Mueller

K. F. MUELLER, Head
Energetic Materials Division

DTIC
COPY
INSPECTED

CONTENTS

ILLUSTRATIONS

SECTION 1

INTRODUCTION

The expansion of a function in terms of orthogonal polynomials can be very useful. These polynomials are easy to manipulate while retaining good convergence properties. The calculation of these polynomials to higher orders is nontrivial and requires the use of a computer in order to obtain reasonable accuracy. These polynomials can then be used to construct a Gaussian integration scheme retaining a degree of precision $2m-1$ where m is the degree of the orthogonal polynomials used. The resulting error in the Gaussian method can be estimated and therefore controlled. These computations extend previous Tables which were compiled by hand calculation.[1,2]

SECTION 2

MATHEMATICAL FORMULATION

2.1 ORTHOGONAL POLYNOMIALS

For every weight distribution there is an associated set of orthogonal polynomials.[3] The polynomials are unique and independent of the choice of constants $a_0, a_1, a_2 \ldots a_n$ which can be given arbitrary nonzero values. For $n>0$ the orthogonal polynomials will satisfy a three-term recursion relationship as follows:

$$\phi_{n+1}(x) = \alpha_n(x-\beta_n)\phi_n(x) - \gamma_n\phi_{n-1}(x)$$

with

$$\phi_{-1}(x)=0, \quad \phi_0(x)=a_0, \quad \alpha_n=a_{n+1}/a_n$$

and

$$\beta_n = \frac{\int x\phi_n(x)\phi_n(x)dx}{\int(\phi_n(x))^2dx}$$

$$\gamma_n = \frac{\int \alpha_n \phi_n(x) \, x \, \phi_{n-1}(x)dx}{\int \phi_{n-1}(x)^2dx}$$

(1)

In general the integrations above can become quite cumbersome and difficult to carry out by hand. However, in the case with weight $\ln(x)$, a relationship can be developed reducing the integration to a constant, dependent only on the power of x. These relationships are:

$$\int_0^1 x^n \ln(x)dx = \frac{(-1)}{(n+1)^2}$$

(2)

$$\int_{-1}^{1} x^n \ln(x)\, dx = \frac{(1+(-1)^n)}{(n+1)^2} \qquad (3)$$

Making use of these relationships, the problem can be broken-down into the manipulation of polynomials in addition, subtraction and multiplication. This type of calculation is well suited for computer programming. For simplicity we will set the a_0, a_1, a_2, a_3, ..., a_n coefficients equal to one. The program can be further simplified through modulation. The final program can calculate orthogonal polynomials, with weight $\ln(x)$, to degree n.

The numerical accuracy of the polynomials is determined by the significant figures retained by the computer. Initially, it is important to retain a high degree of accuracy so that the resulting Gaussian integration scheme will retain accuracy to a significant number of decimal places. This will become more evident as the formulation for the weight factors in the Gaussian integration scheme are developed. For now a 34 decimal place accuracy, which is the limit of VAX FORTRAN Quad precision, is retained. The first four orthogonal polynomials for weight $\ln(x)$ in the interval $0 \le x \le 1$ are:

$\phi_0 = 1$
$\phi_1 = x$ - (1/4)
$\phi_3 = x^2$ - (5/7)x + (17/252)
$\phi_4 = x^3$ - (3105/2588)x² + (178281/501425)x - (4679/258800)

These polynomials are given in decimal form to order ϕ_8 in Appendix A. Using the same nomenclature, the orthogonal polynomials for $\ln[1/|x|]$ in the interval $-1 \le x \le 1$ are:

$\phi_0 = 1$
$\phi_1 = x$
$\phi_3 = x^2$ - (1/9)
$\phi_4 = x^3$ - (9/25)x

These polynomials are also given to order ϕ_8 in Appendix B. From the recursion relationship for ϕ_n, each new polynomial is observed to depend on the accuracy of the previous polynomial. For operations in addition, this will result in the loss of significant figures roughly equivalent to their deviation from unity. This is a factor in the computation over the interval

$0 \leq x \leq 1$ which has higher variations in the polynomial's constants than for the interval $-1 \leq x \leq 1$. Therefore, care was taken in the calculation of the corresponding roots and the weights used in the Gaussian integration scheme to control the roundoff error. The roots of the polynomials were calculated using a standard Newton-Raphson method. The method allowed the accuracy of the roots to be controlled to a specified number of significant figures. Twenty-four decimal places were retained allowing 10 decimal places to be lost in the original computation of the orthogonal polynomials. A higher accuracy in the calculation of the orthogonal polynomials would result in more significant digits in the Gaussian integration scheme, which could be accomplished with some clever programing techniques. However, it was felt for general applications a 16 to 20 decimal place accuracy in the final Gaussian integration would be sufficient. The computer program used in the calculations is provided in Appendix C. The program is capable of calculating orthogonal polynomials weight $\ln(x)$ to order n within the limitations of the computer used.

2.2 GAUSSIAN QUADRATURE

The Gaussian quadrature formulation will be discussed to show how weight and error factors in the Gaussian integration scheme are calculated. The description of the method will show the link between the orthogonal polynomials calculated in Section 2.1 and the resulting Gaussian quadrature formulation. The basic formula for Gaussian integration is:

$$\int_a^b f(x)w(x)dx = \sum_{j=1}^m H_j f(x_j) + \frac{f^{(2m)}(\xi)}{(2m)!} \int_a^b [\pi(x)]^2 w(x) \, dx \qquad (4)$$

where, H_j is the Gaussian integration weight factor and x_j terms are the roots of the orthogonal polynomials, of order m, which were calculated in Section 2.1. The error is a function of the $2m^{th}$ derivative of $f(x)$ and $\pi(x)$ will be given in the Gaussian Quadrature development. The formulation follows the nomenclature given in F. B. Hildebrand's classic book, "Introduction to Numerical Analysis."[4]

The formulation begins by noting that the values of $f(x)$ and its derivative $f'(x)$ are known at m points between a and b in ascending order, $a < x_1 < x_2 < x_3 < \ldots\ldots x_m < b$. The

auxiliary functions:

$$\pi(x) = (x - x_1)(x - x_2).....(x - x_m) \tag{5}$$

and

$$l_i(x) = \frac{\pi(x)}{(x - x_i)\pi'(x_i)} \qquad (i=1,2...m) \tag{6}$$

can now be constructed which have the following properties, $\pi(x_i) = 0$, with $l_i(x_i) = \delta_{ij}$. These important relationships are used to assemble a polynomial of order m-1 which takes on the values of $f(x_1)$, $f(x_2)$ ....$f(x_m)$ in the interval a to b. The resulting expression is written as:

$$y(x) = \sum_{k=1}^{m} l_k(x)f(x_k) \tag{7}$$

The error in the expression has the form:

$$E = \frac{f_{(\zeta)}^{(m)}}{m!}\pi(x) \tag{8}$$

where ζ is in the interval $a < \zeta < b$. Now, taking advantage of the fact that $f(x)$ and $f'(x)$ are known, a polynomial of degree 2m-1 with 2m parameters can be written as:

$$y(x) = \sum_{k=1}^{m} h_k(x)f(x_k) + \sum_{k=1}^{m} \underline{h}_k(x)f'(x_k) \tag{9}$$

where $h_i(x)$ and $\underline{h}_i(x)$ are polynomials of order 2m-1. To satisfy for $y(x_i) = f(x_i)$, the following must hold for $h_i(x_i) = \delta_{ij}$ and $\underline{h}_i(x_i) = 0$. Similarly, for $y'(x_i) = f'(x_i)$, then the values $h_i'(x_i) = 0$, and $\underline{h}'(x_i) = \delta_{ij}$ must hold. Making use of the auxiliary function $l_i(x)$, which is degree m-1, $h_i(x)$ and $\underline{h}_i(x)$ can be written as:

$$h_i(x) = r_i(x) [l_i(x)]^2 \tag{10}$$

and

$$\underline{h}_i(x) = s_i(x) [l_i(x)]^2 \tag{11}$$

These relationships have order 2m-1 and $r_i(x_i)$ and $s_i(x_i)$ are linear functions satisfying $r_i(x_i) = 1$ $r'_i(x_i) + 2l'_i(x_i) = 0$, $s_i(x_i) = 0$ and $s'_i(x_i) = 1$. Combining these expressions yields:

$$h_i(x) = [1 - 2l'_i(x_i)(x - x_i)][l_i(x)]^2 \qquad (12)$$

and

$$h_i(x) = (x - x_i)[l_i(x)]^2 \qquad (13)$$

which with Equation (9) is known as Hermite's interpolation formula. The error associated with Equation (9) is given by:

$$E = \frac{f^{(2m)}(\zeta)}{(2m)!} [\pi(x)]^2 \qquad (14)$$

Now taking $y(x)$ as $f(x)$ the integral is written as:

$$\int_a^b f(x)w(x)dx =$$

$$\sum_{j=1}^m f(x_j) \int_a^b w(x)[1 - 2l'_k(x_k)(x-x_k)][l_k(x)]^2 \, dx +$$

$$\sum_{j=1}^m f'(x_j) \int_a^b w(x)(x-x_k)[l_k(x)]^2 \, dx + \qquad (15)$$

$$\frac{f^{(2m)}(\zeta)}{(2m)!} \int_a^b w(x)[\pi(x)]^2$$

If $\pi(x)$ is orthogonal to $l_1(x), l_2(x) \ldots l_m(x)$ over (a,b) relative to the weighting functions $w(x)$, the second term in Equation (15) will vanish and the resulting expression will reduce to:

$$\int_a^b f(x)w(x)dx = \sum_{j=1}^m H_j f(x_j) + \frac{f^{(2m)}(\zeta)}{(2m)!} \int_a^b [\pi(x)]^2 w(x) \, dx \qquad (16)$$

with

$$H_k = \int_a^b w(x)[l_k(x)]^2 \, dx \qquad (17)$$

while retaining accuracy of 2m-1. Rather than calculating the Gaussian Integration weight factors H_k directly from Equation (17), which could become quite difficult, they can be determined by taking advantage of Equation (16)'s 2m-1 accuracy and allowing $f(x) = x^m$. With m = 0, 1, 2 ... m-1 Equation (16) can be calculated exactly. The result will yield a matrix:

$$A_{ij} \ H_j = \int_a^b x^i \ln(1/|x|)dx = const. \qquad (i=0,1....m-1) \qquad (18)$$

where

$$A_{ij} = \begin{bmatrix} 1 & 1 & 1 & \cdots\cdots\cdots 1 \\ x_1 & x_2 & x_3 & \cdots\cdots\cdots x_m \\ x_1^2 & x_2^2 & x_3^2 & \cdots\cdots\cdots x_m^2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots x_m^{m-1} \end{bmatrix}$$

Equation (18) can be solved yielding the values of the weight H_j using a Gaussian elimination routine. The drawback in this method is the loss in accuracy from the Gaussian elimination. When increased accuracy is required, the Gaussian weight factors can be calculated directly using Equation (17). However, for most applications the above method is sufficient.

SECTION 3

RESULTS

The results from Equation (16) are tabulated in Tables 1 and 2. The numerical accuracy was verified through the calculation of a polynomial of order 2m-1. The Gaussian quadrature should, in this case, be exact. Comparing the Gaussian solution to the exact solution gave an estimation of the total number of significant figures accuracy. As expected, the accuracy was higher for the lower order polynomials than for the higher order polynomials. Further, the accuracy was roughly 16 decimal places in the worst case. Therefore, only the first 16 decimal places are given. All of the polynomials were checked using this procedure.

TABLE 1. GAUSSIAN QUADRATURE ln(x) $0 \leq x \leq 1$

$$\int_0^1 f(x) \text{Log}(1/|x|)dx = \sum_{j=1}^m \alpha_j f(x_j) + \frac{f_{(\xi)}^{(2m)}}{(2m)!}K_m$$

x_i		α_j	
	n=2		
0.11200 88061 66976		0.71853 93190 30384	
0.60227 69081 18738		0.28146 06809 69616	
	n=3		
0.06389 07930 87325		0.51340 45522 32363	
0.36899 70637 15618		0.39198 00412 01488	
0.76688 03039 38941		0.09461 54065 66149	
	n=4		
0.04144 84001 99383		0.38346 40681 45135	
0.24527 49143 20602		0.38687 53177 74763	
0.55616 54535 60276		0.19043 51269 50142	
0.84898 23945 32985		0.03922 54871 29960	
	n=5		
0.02913 44721 51972		0.29789 34717 82894	
0.17397 72133 20898		0.34977 62265 13224	
0.41170 25202 84902		0.23448 82900 44052	
0.67731 41745 82820		0.09893 04595 16633	
0.89477 13610 31008		0.01891 15521 43196	
	n=6		
0.02163 40058 44117		0.23876 36625 78548	
0.12958 33911 54951		0.30828 65732 73947	
0.31402 04499 14766		0.24531 74265 63210	
0.53865 72173 51802		0.14200 87565 66477	
0.75691 53373 77403		0.05545 46223 24886	
0.92266 88513 72120		0.01016 89586 92932	
	n=7		
0.01671 93554 08259		0.19616 93894 25248	
0.10018 56779 15675		0.27030 26442 47273	
0.24629 42462 07931		0.23968 18730 07691	
0.43346 34932 57033		0.16577 57748 10433	
0.63235 09880 47766		0.08894 32271 37658	
0.81111 86267 40106		0.03319 43043 56571	
0.94084 81667 43348		0.00593 27870 15126	

TABLE 1.   (Cont.)

x_i				α_i		
			n=8			
0.01332	02441	60892		0.16441	66047	28003
0.07975	04290	13895		0.23752	56100	23306
0.19787	10293	26188		0.22684	19844	31919
0.35415	39943	51909		0.17575	40790	06070
0.52945	85752	34917		0.11292	40302	46759
0.70181	45299	39100		0.05787	22107	17782
0.84937	93204	41107		0.02097	90737	42133
0.95332	64500	56360		0.00368	64071	04028
			n=9			
0.01086	93360	84175		0.14006	84387	48135
0.06498	36663	38008		0.20977	22052	01030
0.16222	93980	23883		0.21142	71498	96603
0.29374	99039	71675		0.17715	62339	38080
0.44663	18819	05468		0.12779	92280	33205
0.60548	16627	76129		0.07847	89026	11562
0.75411	01371	57164		0.03902	25049	85399
0.87726	58288	35838		0.01386	72955	49593
0.96225	05594	10282		0.00240	80410	36392
			n=10			
0.00904	26309	62200		0.12095	51319	54571
0.05397	12662	22501		0.18636	35425	64072
0.13531	18246	39251		0.19566	08732	77760
0.24705	24162	87160		0.17357	71421	82907
0.38021	25396	09332		0.13569	56729	95484
0.52379	23179	71843		0.09364	67585	38111
0.66577	52055	16425		0.05578	77273	51416
0.79419	04160	11966		0.02715	98108	99233
0.89816	10912	19004		0.00951	51826	02849
0.96884	79887	18634		0.00163	81576	33598
			n=11			
0.00764	39411	74638		0.10565	22560	99100
0.04554	18282	56579		0.16657	16806	00629
0.11452	22974	55125		0.18056	32182	87754
0.21037	85812	27034		0.16727	87367	73784
0.32669	55532	21693		0.13869	70574	01631
0.45545	32469	28813		0.10393	34333	65044
0.58764	83563	59084		0.06953	66978	88735
0.71396	38500	12561		0.04054	16008	03596
0.82545	32178	01812		0.01943	54024	76218
0.91419	39216	12543		0.00673	74293	42450
0.97386	02562	75586		0.00115	24869	61057

TABLE 1.   (Cont.)

x_i				α_i		
			n=12			
0.00654	87222	79080		0.09319	26914	43931
0.03894	68095	60450		0.14975	18275	76322
0.09815	02631	06007		0.16655	74543	64593
0.18113	85815	90632		0.15963	35594	36988
0.28322	00676	67373		0.13842	48318	64836
0.39843	44351	63437		0.11001	65706	35721
0.51995	26267	92353		0.07996	18217	70829
0.64051	09167	16106		0.05240	69548	24642
0.75286	50120	51831		0.03007	10888	73761
0.85024	00241	62302		0.01424	92455	87998
0.92674	96832	23914		0.00489	99245	82322
0.97775	61296	89997		0.00083	40290	38057
			n=16			
0.00389	78344	87115		0.06079	17100	43591
0.02302	89456	16873		0.10291	56775	17581
0.05828	03983	06240		0.12235	56620	46009
0.10867	83650	91053		0.12756	92469	37015
0.17260	94549	09843		0.12301	35746	00070
0.24793	70544	70578		0.11184	72448	55485
0.33209	45491	29916		0.09659	63851	52124
0.42218	39105	81948		0.07935	66643	51473
0.51508	24733	81462		0.06185	04945	81965
0.60755	61204	47728		0.04543	52465	07726
0.69637	56532	28213		0.03109	89747	51581
0.77843	25658	73265		0.01945	97659	27360
0.85085	02697	15391		0:01077	62549	63205
0.91108	68572	22271		0.00497	25428	90087
0.95702	55717	03542		0.00167	82011	10051
0.98704	78002	47984		0.00028	23537	64668
			n=20			
0.00258	83279	57950		0.04314	27521	61381
0.01520	96623	61051		0.07538	37099	48624
0.03853	65503	98586		0.09305	32674	85084
0.07218	16138	58240		0.10145	67118	65901
0.11546	05265	41834		0.10320	17620	51262
0.16744	28563	32738		0.10002	25497	82060
0.22698	37873	09246		0.09325	97992	65015
0.29275	49609	69755		0.08402	89528	32386
0.36327	74298	53964		0.07328	55890	93483
0.43695	71400	46558		0.06185	03368	85688
0.51212	25945	90821		0.05041	66044	21955

TABLE 1.   (Cont.)

x_i	α_i
n=20 (cont.)	
0.58706 40447 84407	0.03955 13700 01102
0.66007 34131 51321	0.02969 40779 02129
0.72948 40837 46511	0.02115 63153 68784
0.79370 96718 02302	0.01412 37329 55045
0.85128 08926 21665	0.00866 09745 19127
0.90087 96807 20293	0.00471 99401 57046
0.94136 97490 37632	0.00215 13974 10105
0.97182 27410 26546	0.00071 97282 17043
0.99153 80814 23101	0.00012 04276 76769

Error Factor

m	K_m
2	2.8527E-03
3	1.7324E-04
4	1.0651E-05
5	6.5868E-07
6	4.0864E-08
7	2.5401E-09
8	1.5809E-10
9	9.8482E-12
10	6.1386E-13
11	3.8281E-14
12	1.4902E-16
16	3.6251E-20
20	5.5140E-25

TABLE 2. GAUSSIAN QUADRATURE ln(x) −1 ≤ x ≤ 1

$$\int_{-1}^{1} f(x)\,Log(1/|x|)\,dx \;=\; \sum_{j=1}^{m} \alpha_j f(x_j) \;+\; \frac{f^{(2m)}(\zeta)}{(2m)!}K_m$$

$\pm x_i$			α_i		
n=2					
0.33333	33333	33333	1.00000	00000	00000
n=3					
0.00000	00000	00000	1.38271	60493	82716
0.60000	00000	00000	0.30864	19753	08641
n=4					
0.21304	15047	38934	0.86489	96815	02982
0.72929	60938	31051	0.13510	03184	97017
n=5					
0.00000	00000	00000	1.09457	98791	69950
0.42048	98338	89206	0.38776	70148	27492
0.80943	17212	07776	0.06494	30455	87532
n=6					
0.15842	27734	26985	0.74816	53867	89280
0.55266	10734	15253	0.21593	27476	27900
0.85720	27159	35678	0.03590	18655	82819
n=7					
0.00000	00000	00000	0.91905	14893	96716
0.32384	19262	08046	0.39366	19293	55519
0.65016	59292	67686	0.12579	98534	11872
0.89012	86734	99548	0.02101	24725	34250
n=8					
0.12670	24568	20194	0.65995	61516	92837
0.44261	10087	71680	0.24698	40796	71079
0.71755	21511	92643	0.07977	87676	46011
0.91233	97218	17349	0.01328	10009	90071
n=9					
0.00000	00000	00000	0.79842	97904	94917
0.26359	77526	12729	0.38007	36092	67535
0.53829	17961	34696	0.15972	57440	76797
0.76891	63270	11850	0.05227	98022	93951
0.92882	94062	74082	0.00870	59491	14257
n=10					
0.10583	33779	87174	0.59201	93756	79453
0.36867	78218	46622	0.25603	33044	69505
0.61013	09472	43461	0.10991	55004	92604
0.80678	01136	00994	0.03604	02553	78796
0.94084	47632	28488	0.00599	15639	79640

TABLE 2.   (Cont.)

$\pm x_i$				α_i		
			n=11			
0.00000	00000	00000		0.70942	89331	43886
0.22243	43034	60040		0.36155	85459	06289
0.45787	79636	99688		0.17673	87580	04816
0.66836	54488	33843		0.07732	25140	30155
0.83674	53191	58890		0.02543	75641	71794
0.95022	91816	34761		0.00422	81513	15001
			n=12			
0.09100	68674	23150		0.53816	26831	93826
0.31583	07442	07888		0.25536	33595	74900
0.52845	71620	84128		0.12833	91477	72704
0.71382	37584	10345		0.05646	84502	49006
0.85989	86117	07467		0.01858	07671	94375
0.95743	35931	44926		0.00308	55920	15185
			n=16			
0.07127	02281	56883		0.45796	46784	03328
0.24549	21041	38222		0.24337	28614	54429
0.41504	86112	42911		0.14477	44109	75228
0.57258	30249	25674		0.08331	93142	30436
0.71203	16306	51971		0.04368	02453	94013
0.82823	11539	92066		0.01942	70191	64386
0.91694	65068	45198		0.00639	99817	92819
0.97496	24353	08112		0.00106	14885	85358
			n=20			
0.05868	47133	89643		0.40077	93096	05514
0.20085	59097	43386		0.22788	83640	17951
0.34102	34666	56996		0.14812	36002	33447
0.47499	75568	52187		0.09654	78165	51015
0.59933	10571	29593		0.06064	97491	22039
0.71098	63017	59170		0.03560	94026	12321
0.80729	48807	41148		0.01880	39939	42658
0.88597	99593	38351		0.00837	97607	34035
0.94519	16589	77233		0.00276	04277	99584
0.98353	84534	54727		0.00045	75753	81431
			n=24			
0.04993	72373	84857		0.35769	00293	66895
0.17001	63351	91168		0.21288	86057	68788
0.28921	06925	78442		0.14615	86826	93945
0.40487	40752	50707		0.10231	58022	81275
0.51485	72812	17607		0.07069	44002	80834
0.61724	49695	71121		0.04728	27060	95115
0.71030	37964	75204		0.03005	40087	21456

TABLE 2. (Cont.)

$\pm x_i$	α_i
n=24 (cont.)	
0.79248 06376 38201	0.01773 83730 49261
0.86241 47885 90768	0.00938 59493 37033
0.91895 35961 89744	0.00418 47379 38628
0.96116 74429 07463	0.00137 83013 39528
0.98836 05896 81911	0.00022 84031 27235
n=28	
0.04349 41764 00850	0.32390 31439 72538
0.14742 92818 32779	0.19930 42967 33486
0.25100 39598 04491	0.14203 37892 50395
0.35241 93609 10315	0.10412 77083 39644
0.45023 89711 34220	0.07627 46613 85082
0.54318 11449 52526	0.05499 11928 92602
0.63006 96133 34514	0.03856 77165 33202
0.70982 40062 92755	0.02599 72305 72777
0.78146 25013 15631	0.01659 02486 69573
0.84410 84878 12482	0.00981 00812 72568
0.89699 83042 16216	0.00519 45033 21041
0.93948 87558 94016	0.00231 62505 40379
0.97106 36416 52642	0.00076 27924 50803
0.99133 76283 29069	0.00012 63840 65902
n=34	
0.03647 71393 63964	0.28480 88700 72993
0.12297 98221 19973	0.18173 17245 19901
0.20946 47888 96709	0.13455 47332 93387
0.29481 97514 16519	0.10319 69140 06127
0.37818 01056 68895	0.07983 50423 79525
0.45877 99845 11221	0.06153 96373 69396
0.53591 10708 28227	0.04689 14786 45456
0.60891 09589 81738	0.03508 51754 12643
0.67716 09074 73224	0.02560 50459 80362
0.74008 70785 20741	0.01808 47481 86851
0.79716 31474 66867	0.01223 88747 32165
0.84791 33925 37128	0.00782 68355 03823
0.89191 58393 78835	0.00463 27525 51770
0.92880 52304 04085	0.00245 39343 26313
0.95827 56693 32595	0.00109 42328 21961
0.98008 27591 90213	0.00036 03099 08124
0.99404 42773 90478	0.00005 96902 89197

TABLE 2.   (Cont.)

Error Factor

	K_m
2	5.5309E-02
3	1.2016E-02
4	3.0609E-03
5	7.1423E-04
6	1.8184E-04
7	4.3511E-05
8	1.1052E-05
9	2.6774E-06
10	6.7860E-07
11	1.6560E-07
12	4.1898E-08
16	1.6110E-10
20	6.2325E-13
24	2.4186E-15
28	9.4030E-18
34	2.2840E-21

REFERENCES

1. Mineur, H., _Techniques de calcul numerique_, Librairie Polytechnique, Charles Beranger, Paris, France, 1952, pp. 555-556.

2. Abramowitz, A., and Stegun, I. A., _Handbook of Mathematical Functions_, Dover Publications, Inc., New York, NY, 1972.

3. AkeBjörck, G. D., _Numerical Methods_, Prentice-Hall, Englewood Cliffs, NJ, 1974.

4. Hildebrand F. B., _Introduction to Numerical Analysis_, Second Edition, Dover Publications, Inc., New York, NY, 1986.

APPENDIX A

ORTHOGONAL POLYNOMIALS WEIGHT $\ln(x)$ $0 \leq x \leq 1$

$$\phi_n = \Sigma\ a_i x^i \qquad a_i\ (i=0,n)$$

n	a_0	a_1	a_2 .....
0	1.00000000		
1	-0.25000000,	1.00000000	
2	0.06746032,	-0.71428571,	1.000000000
3	-0.01807960, 1.00000000	0.35554869,	-1.199768161,
4	0.00480026, -1.69187124,	-0.14966864, 1.00000000	0.885229712,
5	-0.00126470, 1.66127218,	0.05703065, -2.18689974,	-0.514118407, 1.000000000
6	0.00033117, -1.23444544, 1.00000000	-0.02032932, 2.68540097,	0.257587994, 2.683479253,
7	-0.00008630, 0.76866244, -3.18098055,	0.00690690, -2.43476271, 1.00000000	-0.116584253, 3.958396326,
8	0.00002240, -0.42271739, 5.48066379,	-0.0022629, 1.8144789, -3.6790746,	0.048965172, -4.239611389, 1.000000000

APPENDIX B

ORTHOGONAL POLYNOMIALS WEIGHT $\ln(x)$ $-1 \leq x \leq 1$

$$\phi_n = \Sigma\, a_i x^i \qquad a_i \ (i=0,n)$$

n	a_0	a_1	a_2 .....
0	1.00000000		
1	0.00000000,	1.00000000	
2	-0.11111111,	0.00000000,	1.00000000
3	0.00000000, 1.00000000	-0.36000000,	0.000000000,
4	2.41399417, 0.00000000,	0.00000000, 1.00000000	-0.577259475,
5	0.00000000, -0.83199141,	0.11584344, 0.00000000,	0.000000000, 1.000000000
6	-5.63274451, 0.00000000, 1.00000000	0.00000000, -1.06532853,	0.250539503, 0.000000000,
7	0.00000000, 0.46235607, 0.00000000,	-3.51253083, 0.00000000, 1.00000000	0.000000000, -1.319918384,
8	1.34782840, 0.00000000, -1.55920288,	0.00000000, 0.71727267, 0.00000000,	-9.507552587, 0.000000000, 1.000000000

APPENDIX C

COMPUTER PROGRAM

Due to the relative simplicity of the programs, only a few comment cards are included. However, a brief explanation of the program's structure and subroutines will allow modification or improvement for the calculation of a variety of orthogonal polynomials. The programs make use of the recursion relationship in Equation (1) and the integral evaluations given by Equations (2) and (3) of Section 2.1 respectively. POLY1 is more general and calculates both Gama γ and Beta β from Equation (1). However, in POLY2, Beta is zero and therefore not included. Both programs have the three subroutines, POLYMULT, POLYINT, and POLYPLUS. POLYMULT multiplies two polynomials and stores the result in a third polynomial in ascending orders of x. POLYINT integrates a polynomial and returns a real expression in accordance with Equations (2) and (3). POLYPLUS multiplies a polynomial by x. The working polynomials are stored in arrays A through C and the results are stored in ANS. Each polynomial is dependent on the previous polynomial, and the program is looped "n" times to calculate all polynomials to order n+1.

PROGRAM POLY1

```
        PROGRAM POLY1

C       0 < x < 1

        PROGRAM POLY1
        IMPLICIT REAL*16 (A-H,P-Z)
        DIMENSION A(80),B(80),C(80)
        COMMON /AAA/ANS(80,80)
        ANS(1,1)=1.0Q0
        ANS(2,2)=1.0Q0
        ANS(2,1)=-.25Q0
        DO 11 N=2,20
        DO 7 I=1,N+1
        A(I)=0.0Q0
        B(I)=0.0Q0
     7  C(I)=0.0Q0
C
C          FIND A(I)
C
        DO 1 I=1,N
     1  A(I)=ANS(N,I)
        CALL POLYPLUS(A,N)
        CALL BETA(B,N)
        CALL GAMA(C,N)
C
C          CALCULATE POLY
C
        DO 6 I=1,N+1
     6  ANS(N+1,I)=(A(I)-B(I))-C(I)
C
        WRITE(6,*)N+1
        DO 10 I=1,N+1
    10  WRITE(6,*)ANS(N+1,I)
    11  CONTINUE
        STOP
        END
C
C       FIND BETA
C
        SUBROUTINE BETA(E,N)
        IMPLICIT REAL*16 (A-H,P-Z)
        DIMENSION A(80),B(80),C(80),D(80),E(80)
        COMMON /AAA/ANS(80,80)
        DO 1 I=1,N+1
        A(I)=0.0Q0
        B(I)=0.0Q0
```

C-2

PROGRAM POLY1 (Cont.)

```fortran
      C(I)=0.0Q0
    1 D(I)=0.0Q0
      DO 2 I=1,N
      A(I)=ANS(N,I)
      B(I)=A(I)
    2 C(I)=A(I)
      CALL POLYPLUS(A,N)
      CALL POLYMULT(D,B,C,N,N)
      CALL POLYINT(D,DD,2*N)
      CALL POLYMULT(D,A,B,N+1,N)
      CALL POLYINT(D,DD1,2*N+1)
      DO 3 I=1,N
    3 E(I)=(DD1/DD)*B(I)
      RETURN
      END
C
C
C     FIND GAMA
C

      SUBROUTINE GAMA(D,N)
      IMPLICIT REAL*16 (A-H,P-Z)
      DIMENSION A(80),B(80),C(80),D(80)
      COMMON /AAA/ANS(80,80)
      DO 1 I=1,N+1
      A(I)=0.0Q0
      B(I)=0.0Q0
      C(I)=0.0Q0
    1 D(I)=0.0Q0
      DO 2 I=1,N
    2 A(I)=ANS(N,I)
      CALL POLYPLUS(A,N)
      DO 3 I=1,N-1
    3 B(I)=ANS(N-1,I)
      CALL POLYMULT(C,A,B,N+1,N-1)
      CALL POLYINT(C,CC,2*N)
      DO 4 I=1,N+1
    4 A(I)=B(I)
      CALL POLYMULT(C,B,A,N-1,N-1)
      CALL POLYINT(C,CC1,2*(N-1))
      XNUMB=CC/CC1
      DO 5 I=1,N-1
    5 D(I)=ANS(N-1,I)*XNUMB
      RETURN
      END
      SUBROUTINE POLYMULT(A,B,C,K,L)
      IMPLICIT REAL*16 (A-H,P-Z)
      DIMENSION A(80),B(80),C(80)
      DO 1 I=1,K+L
    1 A(I)=0.0Q0
      DO 2 I=1,K
```

PROGRAM POLY1 (Cont.)

```fortran
      DO 2 J= 1,L
    2 A(I+J-1)=A(I+J-1)+B(I)*C(J)
      RETURN
      END
C
C
C     INTEGRATION ROUTINE
C
      SUBROUTINE POLYINT(A,AA,N)
      IMPLICIT REAL*16 (A-H,P-Z)
      DIMENSION A(80)
      AA=0.0Q0
      DO 1 I=1,N-1
    1 AA=AA+A(I)*(-1.0Q0)/(QFLOAT(I)**2)
      RETURN
      END
C
C
C     SHIFTS THE POLY BY ONE
C
      SUBROUTINE POLYPLUS(A,N)
      IMPLICIT REAL*16 (A-H,P-Z)
      DIMENSION A(80),B(80)
      DO 1 I=1,N+1
    1 B(I+1)=A(I)
      DO 2 I=1,N+1
    2 A(I)=B(I)
      RETURN
      END
```

PROGRAM POLY2

```fortran
      PROGRAM POLY2

C     -1 < x < 1

      IMPLICIT REAL*16 (A-H,P-Z)
      DIMENSION A(80),B(80),C(80)
      COMMON /AAA/ANS(80,80)
      ANS(1,1)=1.0Q0
      ANS(2,2)=1.0Q0
      DO 11 N=2,36
      DO 7 I=1,N+1
      A(I)=0.0Q0
      B(I)=0.0Q0
    7 C(I)=0.0Q0
C
C
C        FIND A(I)
C
      DO 1 I=1,N
    1 A(I)=ANS(N,I)
      CALL POLYPLUS(A,N)
      CALL GAMA(C,N)
C
C
C        CALCULATE POLY
C
      DO 6 I=1,N+1
    6 ANS(N+1,I)=A(I)-C(I)
C
      WRITE(6,*)N+1
      DO 10 I=1,N+1
   10 WRITE(6,*)ANS(N+1,I)
   11 CONTINUE
      STOP
      END
C
C
C     FIND GAMA
C
      SUBROUTINE GAMA(D,N)
      IMPLICIT REAL*16 (A-H,P-Z)
      DIMENSION A(80),B(80),C(80),D(80)
      COMMON /AAA/ANS(80,80)
      DO 1 I=1,N+1
      A(I)=0.0Q0
      B(I)=0.0Q0
      C(I)=0.0Q0
    1 D(I)=0.0Q0
      DO 2 I=1,N
```

C-5

PROGRAM POLY2 (Cont.)

```fortran
  2 A(I)=ANS(N,I)
    CALL POLYPLUS(A,N)
    DO 3 I=1,N-1
  3 B(I)=ANS(N-1,I)
    CALL POLYMULT(C,A,B,N+1,N-1)
    CALL POLYINT(C,CC,2*N)
    DO 4 I=1,N+1
  4 A(I)=B(I)
    CALL POLYMULT(C,B,A,N-1,N-1)
    CALL POLYINT(C,CC1,2*(N-1))
    XNUMB=CC/CC1
    DO 5 I=1,N-1
  5 D(I)=ANS(N-1,I)*XNUMB
    RETURN
    END
    SUBROUTINE POLYMULT(A,B,C,K,L)
    IMPLICIT REAL*16 (A-H,P-Z)
    DIMENSION A(80),B(80),C(80)
    DO 1 I=1,K+L
  1 A(I)=0.0Q0
    DO 2 I=1,K
    DO 2 J= 1,L
  2 A(I+J-1)=A(I+J-1)+B(I)*C(J)
    RETURN
    END
C
C       INTEGRATION ROUTINE
C
    SUBROUTINE POLYINT(A,AA,N)
    IMPLICIT REAL*16 (A-H,P-Z)
  . DIMENSION A(80)
    AA=0.0Q0
    DO 1 I=1,N-1
  1 AA=AA+A(I)*(-(1.0Q0+(-1.0Q0)**(I-1))/(QFLOAT(I)**2))
    RETURN
    END
C
C       SHIFTS THE POLY BY ONE
C
    SUBROUTINE POLYPLUS(A,N)
    IMPLICIT REAL*16 (A-H,P-Z)
    DIMENSION A(80),B(80)
    DO 1 I=1,N+1
  1 B(I+1)=A(I)
    DO 2 I=1,N+1
  2 A(I)=B(I)
    RETURN
    END
```

DISTRIBUTION

	Copies		Copies
The Johns Hopkins University Department of Mechanics Latrobe Hall Attn: Dr. A. Prosperetti Dr. O. Hassan Baltimore, MD 21218	 2 1	Internal Distribution: R14 (S. Wilkerson) R14 (K. Kiddy) R14 (J. Shaker) R14 (G. Harris) R14 (H. Huang) R14 (M. Moussouros) R14 (T. Farley) R14 (J. Gaspin) R14 (W. McDonald) R14 (J. Koenig) E231 E232	 20 1 1 1 1 1 1 1 1 1 2 15
NSACSS Attn: G74 (TA) Ft. George G. Meade, MD 20755-6000	 1		
Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	 12		
Library of Congress Attn: Gift and Exchange Division Washington, DC 20540	 4		
Center for Naval Analysis 4401 Fort Avenue P.O. Box 16268 Alexandria, VA 22302-0268	 1		