PE 00562-931-7

Fabry-Perot 간섭계 자료의 새로운 분석방법에 관한 연구

A Study on a New Method of Doppler line profile analysis for Fabry-Perot interferometer data

1996. 3

한 국 해 양 연 구 소

제 출 문

한국해양연구소 소장 귀하

본 보고서를 "Fabry-Perot 간섭계 자료의 새로운 분석방법에 관한 연구" 사업의 최종 보고서로 제출합니다.

1996년 3월

한국해양연구소
연구책임자: 원영인

요 약 문

1. 제 목

Fabry-Perot 간섭계 자료의 새로운 분석방법에 관한 연구

2. 연구개발의 목적 및 중요성

세종기지에서의 고층대기연구는 1989년 시작하여 2년여간 수행되다가 전문인력의 부재로 3년여간 중단되었다. 하지만 1994-95년 하계연구기간중 고층대기연구기기인 Fabry-Perot 간섭계의 점검 및 보정이 이루어져 앞으로 세종기지에서의 고층대기물리 연구는 다시 활발히 이루어질 예정이다. 이에따라 많은 양의 데이터가 축적될 것이며 자료분석에는 자료를 얻는것 이상으로 많은 시간이 소모될 것이다. 현재 남극 세종기지의 Fabry-Perot 간섭계 데이터 분석에 쓰이는 프로그램은 1970년대에 짜여진 것으로 프로그램 언어(BASIC)도 현재 잘 쓰여지지않는 것일 뿐 아니라 분석에도 상당한 시간이 소요되는 문제점을 않고 있다. 이러한 문제점은 앞으로 데이터 양이 많아질수록 더욱 심각하게 될 것으로 여겨지며 따라서 보다 효율적인 소프트웨어의 개발이 시급하다.

3. 연구 내용 및 범위

본 연구는 Fabry-Perot 간섭계로부터 얻어진 자료를 보다 신빙성 있고 빠르게 분석하는 방법을 찾는데 목표를 두고 있으며 이를 위해 여러가지 분석방법을 시도하였다. 방법은 기본적으로 다중회귀(multiple regression)을 이용한 최소자승 근사(least-square fitting)이며 매개변

수(parameter)의 최적값을 찾아가는 방법으로 격자탐색(grid search), 경도탐색(gradient search), 그리고 근사함수의 선형화(linearization) 등의 방법을 시도하였다.

- 비선형함수의 다중회귀 (multiple regression)분석 방법모색
- 다양한 분석방법을 시도
- 채택된 방법들의 비교분석
- 최적 분석방법 선별
- 효율적 자료분석 소프트웨어의 개발

4. 연구결과 및 활용방안

- 각 분석방법의 장,단점 파악
- 새로운 자료분석방법시도
- 간섭계 자료처리에 응용
- 분석과정에 따르는 결과의 불확실성 추정

SUMMARY

1.Title

A study on a new method of Doppler line profile analysis for Fabry-Perot interferometer data

2. Purpose and Importance of the Study

The Fabry-Perot interferometer has been in operation since January 1989 until 1991. The FPI operation program has been resumed in winter of 1994 and two trips have been made since then to remedy the system. One visit in the winter of 1994 was aimed for re-aligning of the system. We planned to replace the He-Ne laser and some other old parts of the FPI system in the end of 1996 or in the early of 1997 and then to perform the FPI operation. As a consequence, much data are expected to come out. The data analysis program that is used now was originally coded in 1970 using BASIC language. This routine is inefficient and inconvenient to use for today's computer. This problem will become serious as more data would come out in near future and there should be a way to improve the software. For this purpose, a new method of data anaysis is tested and will be applied to Fabry-Perot data.

3. Contents and Scope of the Study

There are various ways to analyze observed data which is not approximated by linear function. Those least-square fitting methods include grid search, gradiant search and linearization of function. The three methods listed above have been tested and compared. The most efficient method was adopted for the future usage.

4. Results and Discussions

- comparison of each analysis method
- application of new method to Fabry-Perot data
- determination of error resulted from the analysis

CONTENTS

LIST OF TABLES

LIST OF FIGURES

목차

1. 서론

세종기지에서의 고층대기연구는 1989년 시작하여 2년여간 수행되다가 전문인력의 부재로 3년여간 중단되었다. 하지만 1994-95년 하계연구기간중 고층대기연구기기인 Fabry-Perot 간섭계의 점검 및 보정이 이루어져 앞으로 세종기지에서의 고층대기물리 연구는 다시 이루어 질 예정이다. 이에따라 많은 양의 데이터가 축적될 것이며 자료분석에는 자료를 얻는것 이상으로 많은 시간이 소모될 것이다. 현재 남극 세종기지의 페브리-페로 간섭계 데이터 분석에 쓰이는 프로그램은 1970년대에 짜여진 것으로 프로그램 언어(BASIC)도 현재 잘 쓰여지지않는 것일 뿐 아니라 분석에도 상당한 시간이 소요되는 문제점을 않고 있다. 또한 타 기종 컴퓨터와의 호환송 등 많은 문제점을 내포하고 있다. 이러한 문제점은 앞으로 데이터 양이 많아질수록 더욱 심각하게 될 것으로 여겨지며 따라서 보다 효율적인 소프트웨어의 개발이 시급하다. 따라서 본 연구에서는 페브리-페로 간섭계로부터 얻어진 자료를 보다 신빙성있고 빠르게 분석하는 방법을 모색하는데 그 목표를 두고 있고 이를 위해 몇 가지 알려진 분석방법을 시도하였다. 분석방법은 기본적으로 다차원 공간에서의 비선형함수 최소자승법을 이용하며 알려진 방법중 격자탐색(grid search), 경도탐색(gradient search), 그리고 근사함수의 선형화(linearization)방법들을 이용하며, 각 방법들의 결과를 비교분석하여 가장 안정되고 효과적인 방법을 선택할 것이다. 아울러 선택된 방법은 시험적으로 페브리-페로 간섭계 자료 분석에 응용시켜 추후 본격적인 자료분석에 활용할 계획이다.

2. 본론

때때로 우린 관측으로 얻은 자료 y_i를 변수 x의 함수로 나타내고 싶을때가 있다. 즉 $y=f(x)$라는 함수 f 를 찾고싶을 때가 있는데 여기서 함수 f는 $y(x)=a_1\sin(a_2x)$ 나 $y(x)=a_1x+a_2x+a_3e^{a_4x}$ 등 어떤 함수일 수도 있다. 일반적으로 함수 f 또는 매개변수 a_j를 찾는 방법으로 최소자승 최적화(least-square fitting)방법이 사용되고 있다. 최소자승 χ^2는 다음과 같이 표현되며

$$\chi^2=\sum\left\{\frac{1}{\sigma_i^2}[y_i-y(x_i)]^2\right\} \qquad \text{II-1}$$

여기서 y_i 는 관측된 자료값, $y(x_i)$는 x_i에대한 함수값, 그리고 σ_i는 관측자료의 불확정성 정도를 나타낸다.

그리고 난후 최소자승방법에 의해 매개변수 a_j의 최적값은 각각의 매개변수 a_j에 대해 χ^2가 최소가 되게하는 방법으로 얻어진다.

$$\frac{\partial}{\partial a_j}\chi^2=\frac{\partial}{\partial a_j}\sum\left\{\frac{1}{\sigma_i^2}[y_i-y(x_i)]^2\right\} \qquad \text{II-2}$$

하지만 함수 y(x)가 비선형일 경우 매개변수 a_j를 구하기 위해 수식을 유도하는 것은 그리 쉽지가 않다. 이런 경우 χ^2 를 n개의 매개변수 a_j에 대한 연속함수로 가정하고 n차원의 공간에서(예, 2차원공간; 그림1) χ^2가 최소화 되는 곳을 찾아가면 된다. 이러한 탐색(search)과정의 어려움의 하나는 χ^2가 최저점이 되는 곳이 곳곳에 있을 수 있다는 사실이다. 최저점이 오로지 하나만 존재하는 일정지역으로 범위를 제한하지 않는 한 매개변수 공간(parameter space)을 전체적 최저점을 포함하도록 나눈 다음 매개변수의 범위를 정하여 최저점을 찾아가도록 하는 것이 좋다.

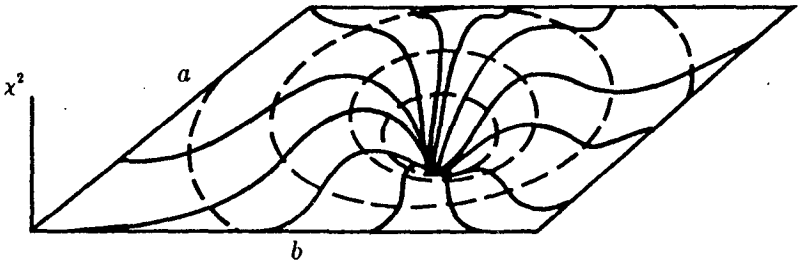


Figure 1 Representative hypersurface describing variation of χ^2 vs. two parameters a and b (Bevington, 1969).

가장 간단한 탐색방법으로는 매개변수 a_j를 n개의 일정한 증가량 Δa_j로 나누어 각각의 hypercube의 정점에서 χ^2의 수치를 구하는 방법이다. 또 다른 방법으로는 충분히 많은 시작점에서 탐색을 시작하여 지역적 최소값들을 대부분 찾아내면 된다.

상기에서 기술되었듯이 χ^2의 최소값을 찾아가는 방법은 다양하며 본 연구에서는 매개변수 공간에서의 탐색방법과 근사분석방법 (approximate analytical methods)에 의한 최소자승방법을 각기 시도하였다. 매개변수공간에서의 탐색방법으로는 격자탐색 (grid search)과 경도탐색(gradiant search)방법이 시도되었고 분석적 방법으로는 함수 f(x)를 선형식으로 전개하여 선형식에 대한 최소자승법을 이용하였다.

2-1. 격자탐색(Grid Search)

χ^2의 변화가 각각의 매개변수 a_j에 대하여 독립적이라고 가정한다면 매개변수의 최적값은 단순히 각각의 매개변수들에 대해 χ^2를 최소로 하는 방법에 의해 구할 수 있다. 이러한 방법를 격자탐색이라고 한다. 각각의 매개변수에 대해 계속되는 반복(iteration)으로 지역적 최소값을 구하고

나서 절대적 최소치를 원하는 정확도에 따라 구할 수가 있다. 그림 2의 예를 보면 χ^2의 등치선이 두가지 매개변수에 대해 나타남을 볼 수 있으며 이 경우 격자탐색은 한쪽 끝에서 시작하여 지그재그 방식으로 타원형의 최소치로 찾아감을 볼 수 있다. 그림에서 보이듯이 격자탐색방법은 최소값으로 수렴해가는 과정이 다소 효과적이지 못하지만 계산이 간단하다는 잇점이 있다.

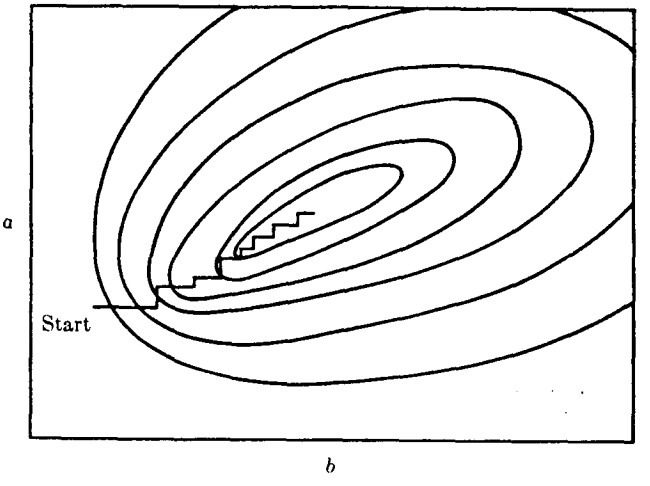


Figure 2. Tortuous path of gird search in two parameter space (Bevington, 1969).

격자탐색은 다음과 같은 절차로 수행된다.

1) 한개의 매개변수 a_j를 $\chi 2$가 감소하는 방향으로 Δa_j만큼 변화시킨다.

2) 매개변수 a_j를 $\chi 2$가 다시 증가할 때 까지 반복적으로 Δa_j만큼 변화시킨다.

3) 최소치 부근에서의 $\chi 2$의 변화가 매개변수 a_j의 포물선으로 기술된다고 가정하고 마지막 3개의 a_j값에 의한 $\chi 2$값을 이용, 최소치를 찾는다.

$$a_j(3) = a_j(2) + \Delta a_j(1) + 2\Delta a_j$$

$$\chi^2(3) > \chi^2(2) \leq \chi^2(1)$$

4) $\chi 2$가 최소치가 되는 곳의 매개변수 값은 다음과 같이 주어진다.

$$a_j(\min) = a_j(3) - \Delta a_j\left[\frac{\chi^2(3)-\chi^2(2)}{\chi^2(3)-2\chi^2(2)+\chi^2(1)}+\frac{1}{2}\right] \qquad \text{II-3}$$

5) 나머지 매개변수에 대해서도 위와같은 방법으로 $\chi 2$를 최소화시키는 값을 구한다.

6) 위와 같은 방법을 지속적으로 반복하여 $\chi 2$의 감소가 더이상 진전되지 않을때까지 수행한다.

2-2. 경도탐사(Gradient Search)

그림 2에서 나타난 것처럼 탐색이 지그재그 방향으로 진행하지 않고 보다 직접적으로 최저점에 도달하는 방향으로 진행된다면 탐색과정이 보다 향상될 것이다. 최소자승 방법중 경사탐사(gradient search)에서는 각각의 매개변수 aj가 제 각각으로 수행하는 격자탐색의 번거로움과 달리 모든 매개변수 a_j가 $\chi 2$의 최대변화방향으로 동시에 변화하며 최적값을 찾아가는 방법이다.

여기서 경사도 $\nabla\chi^2$은 벡터량으로서 각각의 성분들은 $\chi 2$가 최대로 변화하는 방향으로의 변화율이 된다.

$$\nabla\chi^2 = \sum_{j=1}^{n}\left[\frac{\partial\chi^2}{\partial a_j}\hat{a}_j\right] \qquad \text{II-4}$$

위에서 $\hat{a}_j$는 단위행렬을 의미하고 경사도를 구하기 위해서는 시작점 주위에서의 $\chi 2$의 변화가 각 매개변수에 상관없이 추출되어 대략적인 초기 기울기를 구한다.

$$\left(\nabla\chi^2\right)_j = \frac{\partial\chi^2}{\partial a_j} \cong \frac{\chi^2\left(a_j + f\Delta a_j\right) - \chi^2\left(a_j\right)}{\cdot\, f\Delta a_j} \qquad \text{II-5}$$

이 기울기를 구하기 위해 변화되는 a_j의 양은 매개변수의 계단규모(step size) Δa_j 보다는 작아야 하며 분할(fraction) f는 약 10%(f=0.1) 정도의 값을 주게 된다.

실제적으로 각각의 매개변수 a_j는 그 차수(dimension)가 같지 않기 때문에 각 경사도의 차수는 갖지 않을 수 있다. 이런 이유로 차수가 없는 (dimensionless) 매개변수 b_j를 다음과 같이 정의하며

$$b_j = \frac{a_j}{\Delta a_j} \qquad \text{II-6}$$

역시 차수가 없는 크기 1인 경사도를 아래와 같이 정의할 수 있다.

$$\gamma_j = \frac{\partial \chi^2 / \partial b_j}{\sqrt{\sum_{k=1}^{n} \left(\frac{\partial \chi^2}{\partial b_k} \right)^2}} \qquad \frac{\partial \chi^2}{\partial b_j} = \frac{\partial \chi^2}{\partial a_j} \Delta a_j \qquad \text{II-7}$$

경사도탐사는 $\chi 2$의 감소가 가장 큰 방향으로 향하며 각 매개변수의 변화량은 차수없는 경사도 γ과 Δa_j에 의해 결정된다.

$$\delta a_j = -\gamma_j \Delta a_j \qquad \text{II-8}$$

여기서 '-' 부호는 최소자승 $\chi 2$가 감소하는 방향을 의미한다.

경사탐사를 계속적으로 수행하는 방법에는 몇가지 선택이 있을 수 있는데 적당한 방법의 하나는 $\chi 2$의 값이 감소하다가 다시 증가할때까지 처음의 경사방향으로 진행하는 것이다. 일단 $\chi 2$의 값이 다시 증가하게 되면 그곳에서 다시 경사도를 계산하여 새로운 방향으로 탐색을 진행하게 된다. 언제나 탐사가 최소값 부근에 다다르게 되면 타원내삽(parabolic interpolation)을 통해 $\chi 2$의 최소위치를 찾는데 정확도를 높인다.

좀 더 세련된 방법으로는 유한편차(finite difference)를 이용하여 식 II-5에서 구하는 $\chi 2$의 2차 편미분을 이용하는 방법이 있다.

$$\left. \frac{\partial \chi^2}{\partial a_j} \right|_{a_j + \Delta a_j} \cong \left. \frac{\partial \chi^2}{\partial a_j} \right|_{a_j} + \sum_{k=1}^{n} \left(\frac{\partial^2 \chi^2}{\partial a_j \partial a_k} \delta a_k \right) \qquad \text{II-9}$$

이렇게 구해지는 2차 편미분은 탐사경로의 기울기를 수정하며 탐사가 최소값에 근사하게 되면 반복되는 계산을 줄이는 효과가 있다. 하지만 탐색

이 최저치 부근에 충분히 도달하지 않으면 오히려 불필요한 계산과정을 증가시킬 수 있기 때문에 본 연구에서는 2차미분의 사용을 시행하지 않았다.

2-3. 함수의 선형화 (Linearization of function)

분석적 방법을 이용하여 $\chi 2$를 기술하는 방법의 하나는 근사함수 y(x)를 매개변수 a_j에 대해 급수전개하여 선형화(linearization) 시킨후 선형최소자승법을 이용하는 것이다. 이렇게 되면 분석적 기술이 보다 용이해 질 수 있다.

2-3-1. 1차 확장(first-order expansion)

우선 근사함수 y(x)를 매개변수 a_j에 대해 일차식으로 급수전개시키면 다음과 같이 나타낼 수 있다.

$$y(x) = y_0(x) + \sum_{j=1}^{n} \frac{\partial y_0(x)}{\partial a_j} \delta a_j \qquad \text{II-10}$$

결과적으로 급수전개된 함수는 매개변수 증가량 δa_j에 대해 선형이 되며 따라서 선형 최소자승 방법을 이용할 수 있다.

위 식에서 미분값은 시작점 $y_0(x)$에서 구하게 되며, 실제적인 계산은 분석적인 방법이나 수치적 방법을 이용할 수 있다. 수치적 계산은 다음과 같이 행해진다.

$$\frac{\partial y_0(a_j)}{\partial a_j} \equiv \frac{y_0(a_j + \Delta a_j) - y_0(a_j - \Delta a_j)}{2\Delta a_j} \qquad \text{II-11}$$

그리고 난 후 최소자승 χ^2는 매개변수 증가량 δa_j에 대해 다음과 같이 명확한 수식으로 나타낼 수 있다.

$$\chi^2 = \sum \left(\frac{1}{\sigma_i^2} \left\{ y_i - y_0(x_i) - \sum_{j=1}^{n} \left[\frac{\partial y_0(x_i)}{\partial a_j} \delta a_j \right] \right\}^2 \right) \qquad \text{II-12}$$

여기서 y_i'를 다음과 같이 정의하면

$$y'(x) = \sum_{j=1}^{n} \left[\frac{\partial y_0(x_i)}{\partial a_j} \delta a_j \right] \qquad \text{II-13}$$

각 자료는 함수 $y_i'(x)$를 이용, 선형적으로 근사시킬 수 있으며 이때 매개변수 $a_j' = \delta a_j$ 그리고 근사함수는 아래와같이 나타내어 진다.

$$X_j(x_i) = \partial y_0(x_i) / a_j \qquad \text{II-14}$$

이후의 과정은 선형 최소자승법을 이용, δa_j에 대해 χ^2를 최소화시키면 된다.

$$\frac{\partial \chi^2}{\partial \delta a_k} = -2 \sum \left(\frac{1}{\sigma_i^2} \left\{ y_i - y_0(x_i) - \sum_{j=1}^{n} \left[\frac{\partial y_0(x_i)}{\partial a_j} \delta a_j \right] \right\} \frac{\partial y_0(xi)}{\partial a_k} \right) \qquad \text{II-15}$$

$$= 0$$

위 식으로부터 다음과 같이 n개의 방정식이 도출된다.

$$\beta_k \equiv -\frac{1}{2} \frac{\partial \chi_0^2}{\partial \delta a_k} = \sum_{j=1}^{n} \left(\delta a_j \alpha_{jk} \right) \qquad k = 1, n$$

$$\beta = \delta a \alpha$$

$$\alpha_{jk} \equiv \sum \left[\frac{1}{\sigma_i^2} \frac{\partial y_0(x_i)}{\partial a_j} \frac{\partial y_0(x_i)}{\partial a_k} \right] \qquad \text{II-16}$$

분석적 방법에 내재하는 단점중의 하나는, 비록 χ^2의 최소치로의 수렴이 빠르게 진행되기는 하지만, 만약 χ^2의 hypersurface가 포물선 형태를 띄는 지역 밖에서 탐색이 시작된다면 충분히 신뢰할 수 없는 결과가 발생할 수 있다는 사실이다. 이와는 반대로 앞에서 기술된 경도탐색은 비록 최소치에 수렴하는 과정이 빠르지는 못하지만 어느 곳에서 시작하건 보다 안정되게 최소치로 접근해 간다. 따라서 경도 탐색과 분석적 방법의 장점을 혼합한 형태의 알고리즘을 이용한다면 보다 이상적인 분석방법이 될 것이다.

2-3-2. 경사 전개(gradient-expansion)

경사탐색과 함수의 선형화를 조합한 알고리즘(Marquardt,1963)은 곡률 행렬(curvature matrix) α의 대각행렬을 λ만큼 변화시킴으로써 얻어질 수 있다.

$$\beta \equiv \delta a \alpha'$$

$$\alpha'_{jk} = \alpha_{jk}(1+\lambda) \quad \text{for } j=k$$

$$\alpha_{jk} \quad \text{for } j \neq k \qquad \text{II-17}$$

만약 λ가 매우 작은 값을 갖는다면 식 II-17은 1차급수와 비슷한 식이 되며 λ가 큰 값을 갖게 되면 곡률행렬의 대각항들이 우세해 져서 결국 행렬식은 n개의 방정식이 된다.

$$\beta_j \approx \lambda \delta a_j \alpha_{jj} \qquad \text{II-18}$$

매개변수 증감량 δa_j의 해는 역행렬식에 의해 다음과 같이 나타내 지며

$$\delta a_j = \sum_{k=1}^{n} \left(\beta_k \varepsilon'_{jk} \right) \qquad \text{II-19}$$

여기서 β_k는 수식 II-18에 나타나 있고 ε'는 행렬 α'의 역행렬이 된다.

factor λ는 분석적 방법의 잇점을 살리기 위해 충분히 작게 선택되야 하며 동시에 χ^2가 감소할 수 있도록 어느정도는 큰 값을 가져야 한다.

Marquardt(1963)에 의해 제안된 문제해결의 방법은 다음과 같다.

1) $\chi^2(a)$를 계산한다.

2) λ=0.001 로 시작한다.

3) 위에 선택한 λ값으로 δa와 $\chi^2(a+\delta a)$를 계산한다.

4) 만약 $\chi^2(a+\delta a)$가 $\chi^2(a)$보다 크면 λ를 10배 크게하고 3번을 다시 수행한다.

5) $\chi^2(a+\delta a)$가 $\chi^2(a)$보다 작으면 λ를 1/10로 줄이고 $a'=a+\delta a$를 새로운 시작점으로 해서 a'를 a로 치환하고 3번으로 돌아가 수행한다.
매번 수행시마다 δa_j를 다시 계산하여 λ를 최적화 할 필요가 있다. 행렬 α_{jk}와 β_j는 매 반복시마다 한번만 계산한다.

본 연구의 선형화 방법은 2-3-2에서 설명된 경사전개 방법을 따르기로 하였다.

3. 결과 및 분석

3-1. 테스트 1

각기 다른 최소자승법에 대한 간단한 시험결과가 표 1에 나타나 있다. 우선적으로 임의의 함수 $y=a_1\cos(a_2x)$, $a_1=3$, $a_2=0.5$ 를 $x=1,24$에 대해 계산하고 약간의 error를 추가한 결과를 다시 $y=a_1\cos(a_2x)$의 함수를 이용 2장에서 기술된 3가지 방법에 의해 근사시켜 보았다. 각 결과는 시작점을 $a_1=2$, $a_2=0.4$로, 증감량은 각기 0.1, 0.05로 일정한 상태로 하고 시행된 결과이다.

탐사방법	χ^2	a_1	a_2
격자탐색	5.82	2.00	0.40
	2.90	0.82	0.49
	0.04	3.06	0.49
	0.04	3.06	5.08
	0.04	3.07	5.01
경도탐색	5.82	2.00	0.40
	0.71	1.99	0.50
	0.71	1.99	0.49
	0.70	2.00	0.49
	0.49	2.37	0.48
	0.32	2.38	0.50
	0.32	2.38	0.50
선형화방법	5.82	2.00	0.40
	2.67	0.97	0.48
	2.11	2.91	0.55
	0.27	2.53	0.50
	0.04	3.07	0.50
	0.04	3.07	0.50

Table.1. An example of comparison of three methods to function values of $y=a_1\cos(a_2x)$, $a_1=3$, $a_2=0.5$

전체적으로 보면 매개변수의 입력값이 참값과 큰 차이를 보이지 않아 수렴속도가 대체로 빠르게 나타나는 것을 볼 수 있다. 우선 격자탐사법을 보면 첫번째 iteration에서 매개변수 a_1의 값이 오히려 크게 벗어났다가 다시 참값으로 수렴하는 것을 볼 수 있고 두번째 iteration 이후로는 최소자승이 별로 변하지 않음을 볼 수 있다. 이에 반해 경도탐사에서는 매개변수값이 꾸준히 근사치에 수렴하는 모습을 보이며 다른 두가지 방법보다 오랜기간 iteration 함을 보인다. 6번째 iteration이후로 최소자승의 변화는 별로 보이지 않고 있으나 여전히 매개변수중 a_1은 참값에서 약간 벗어나 있음을 볼 수 있다. 마지막으로 선형화 방법을 보면 초기에 약간의 편차가 생긴이후로 급속히 참값에 수렴하는 모습을 보이고 있다. 이와같은 아주 간단한 시험결과에서는 각 방법의 차이점이 뚜렷이 나타나지는 않았다. 다음으로 함수를 조금 더 복잡한 형태로 변화시켜 보았다.

3-2. 테스트 2

두번째 테스트에서는 함수를 $y=a_1\cos(a_2x)+a_3x^2$, $a_1=3$, $a_2=0.5$, $a_3=2$, $x=1,24$로 계산한 다음 이 결과값에 테스트 1과 같은 방법으로 근사시켜 보았다. 근사함수가 복잡해 짐에 따라 각 탐사방법의 결과가 어떻게 변하는지 표 2에 나타나 있다.

표 2의 결과는 표 1과 비교하여 몇가지 다른 양상을 보이고 있다. 격자탐색의 경우 수렴하는 정도에 있어 비교적 빠르게 진행되는 것에는 첫번째 결과와 별 차이가 없지만 매개변수중 a_1과 a_2의 값이 참값과는 ($a_1=3$, $a_2=0.5$) 벗어난 다른 값으로 수렴된 결과를 나타내고 있고, 이런 이유로 인해 χ^2도 상대적으로 큰 값을 나타내 근사값의 신뢰도를 떨어뜨리고 있다. 두번째 방법인 경도탐색의 경우 첫번째 테스트보다 수렴속도가 오히려 향상된 면을 보이고 있으나 역시 격자탐색과 마찬가지로 매개변수중 a_1값이 참값인 3과는 차이를 보이고 있다. 하지만 나머지 매개변수

값은 예정된 값으로 빠르게 수렴한 것을 볼 수가 있다. 마지막으로 선형화 방법의 경우 수렴값과 참값이 거의 일치함을 볼 수 있다. 수렴되는 속도 또한 빠르게 진행됨을 볼 수 있다.

탐사방법	χ^2	a_1	a_2	a_3
격자탐색	263.2	2.00	0.40	3.00
	134.5	-3.84	0.37	2.02
	4.95	-2.74	0.32	1.97
	4.76	-7.54	0.31	1.97
	4.75	-8.46	0.30	1.97
	4.75	-8.49	0.30	1.97
경도탐색	263.2	2.00	0.40	3.00
	1.24	1.99	0.53	1.99
	0.61	1.99	0.50	1.99
	0.58	1.99	0.50	2.00
	0.58	1.99	0.50	2.00
선형화방법	263.2	2.00	0.40	3.00
	2.49	0.99	0.48	1.99
	2.06	2.83	0.54	2.00
	0.20	2.89	0.50	2.00
	0.00	2.99	0.50	2.00
	0.00	2.99	0.50	2.00

Table.2. Same as Table 1, but for function values of $y=a_1\cos(a_2x)+a_3x^2$, $a_1=3$, $a_2=0.5$, $a_3=2$

위의 두가지 시험결과를 놓고 분석해 볼때 수렴하는 속도와 수렴된 값의 신뢰도에서 선형화 방법이 가장 만족할 만한 방법으로 판단된다. 물론 위의 두가지 테스트결과만을 가지고 각 방법의 장단점을 결론짓는 것에는 무리가 있으나 일단 선형화 방법의 신뢰도는 만족할 만한 수준이라 생각되며 기존의 자료분석방법에 선형화 분석방법을 시도해 다음장에 간단한 시험결과를 나타내었다.

3-3. Fabry-Perot 자료분석 테스트

페브리-페로 자료의 분석에 앞서 관측된 자료가 어떤 내용을 지니고 있는지 간단히 살펴본다. 본 자료는 Image Plane Detector라고하는 12 채널의 탐기지에서 나오는 자료로 간섭계를 거친 간섭무늬를 증폭시켜 얻은 결과이다(Killeen and Roble, 1984). 각 채널에서 관측되는 신호는 다음과 같이 표현된다.

$$N_i = \frac{A_o \Omega_{it} t Q_i T_{oi} \cdot 10^6}{4\pi} \int_0^\infty T_F(\lambda)\psi(\lambda,\theta_i)Y(\lambda)d\lambda + B_i \qquad \text{III-1}$$

여기서 적분의 앞부분은 광학기구의 특성을 나타내는 계수들로 A_0는 etalon의 넓이, Ω_i 는 field of view, T_{0i}는 기기의 광투과정도, Q_i는 i번째 채널의 광효율(quantum efficiency)을 나타내고 $T_F(\lambda)$는 필터의 투과함수, $\psi(\lambda,\theta_i)$는 기기함수, 그리고 $Y(\lambda)$는 방출선의 선스펙트럼을 나타낸다. 이 선 스펙트럼은 다음과 같이 두 부분으로 나뉘는데

$$Y(\lambda) = \frac{\Re_o \exp -\left(\frac{\lambda-\lambda_1}{\Delta\lambda_1}\right)^2}{\sqrt{\pi}\Delta\lambda_T} + \left.\frac{\partial\Re}{\partial\lambda}\right|_o \qquad \text{III-2}$$

여기서 R_0는 표면밝기, 그리고 λ_1은 도플러 편향이고, $\Delta\lambda_T$는 열폭(thermal width)으로서 아래의 수식으로 나타내 진다.

$$\Delta\lambda_T = \left(\frac{2kT}{m}\right)^{1/2}\frac{\lambda_1}{c} \qquad \text{III-3}$$

수식 III-2의 첫번째 항은 표면밝기의 가우스분포이고 두번째 항은 연속선 밝기를 나타낸다.

기기 함수(instrument function)는 실험적으로 푸리에급수로 표현되며(Killeen and Roble, 1984), 식 III-1은 다시 다음과 같이 표현될 수 있다.

$$Ni = C_{0i}T_{F0}R_0\sum_{i=0}^{16}\left\{\begin{array}{l} a_{ni}\exp[-n^2G^2(T)]\cos n(X+\frac{2\pi M_0 v}{c}) \\ +b_{ni}\exp[-n^2G^2(T)]\cos n(X+\frac{2\pi M_0 v}{c}) \end{array}\right\} + a_{oi}B_{BG} + B_i \qquad \text{III-4}$$

$$G(T) = \frac{\pi}{c}\sqrt{\frac{2kT}{m}}\frac{\lambda_l}{\Delta\lambda_0} = \frac{\pi\Delta\lambda_T}{\Delta\lambda_0} \qquad \text{III-5}$$

$$X = \frac{2\pi(\lambda_0 - \lambda_l + \phi_i)}{\Delta\lambda_0} \qquad \text{III-6}$$

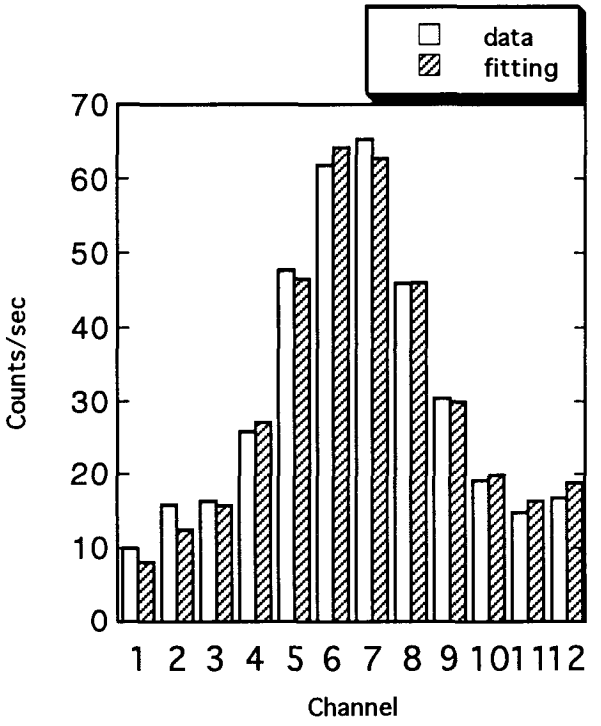
$$B_{BG} = C_{0l}\overline{\Delta\lambda_F}\frac{\partial R}{\partial\lambda}\left(\frac{1-R}{1+R}\right) \qquad \text{III-7}$$

여기서 M_0는 간섭차수의 정수부분, ϕ_i는 i번째 채널의 위상차, B_i는 열적 발산에 의한 배경강도, 그리고 $\Delta\lambda_F$는 필터의 파장폭을 나타낸다.

결국 위의 수식을(III-4) 미지의 표면밝기 X(III-6),온도 T, 속도 v, 연속배경선 B_{BG}, 네개의 독립매개변수를 이용, 관측된 자료에 근사시키는 것이 본 연구의 주안점이다. 본 연구에서는 3절에서 논의된 대로 선형화방법을 이용하였으며 사용된 프로그램은 마지막의 붙임에 나타나 있다.

그림 3은 탐사결과의 한 예이며, 간섭계에서 관측된 자료의 profile N_i(관측된 counts)가 선형화 방법에 의해 온도, 바람, 표면밝기 및 배경선 등의 매개변수로 근사된 결과를 보여주고 있다. 자료는 1988년 12월 3일 북극 그린랜드의 Thule 라는 지역에서 관측된 자료이며 그림에서 빗금친 나무막대표시가 근사시킨 결과를 나타내고 있는데 5번의 iteration으로 충분히 수렴함을 보이고 있다. 여기서 수렴된 결과의 온도는 절대온도(K)이며 입자의 속도는 어떤 기준치에 대한 상대적인 값으로 일반적으로 수직방향의 움직임이 거의 없다는 가정하에 수직방향의 관측값을 기준으로 하여 실제 속도가 결정된다.

Fabry-Perot 630 nm Fringe Profile on 12 channel IPD



	χ^2	온도 (K)	속도	표면밝기 R/Å	배경 R/Å
격자탐색	1119	1000.	-3300.	400.	10.
	205.6	3020.	-1768	2.3	1.7
	203.9	1094.	-1896	2.2	1.2
	29.3	1439.	-1790	2.7	0.7
	11.6	1391.	-1804	3.2	0.7
	11.4	1397.	-1803	3.2	0.7

Figure 3. An example of the iteration technique using linearization of chi-square method in the analysis of FPI acquired fringe profile from Thule, Greenland

관측결과로 나타난 열권의 온도는 약 1400K 정도로 89년 겨울철의 태양활동이 활발했던 것으로 미루어 적당한 결과값이며, 이러한 관측은 날씨 및 태양고도각등의 조건이 만족되면 하루저녁에 약 200회이상 가능하다. 또한 관측결과중 표면밝기 및 배경밝기는 절대적 수치가 아닌 상대적인 세기로, 광도계(photometer)를 이용, 보정한다면 절대적 수치로 환산할 수 있다.

위의 분석방법은 또 다른 관측자료에서도 만족할 만한 결과를 얻을 수 있었으며 함수의 선형화 방법을 이용한 자료의 근사방법이 성공적으로, 그리고 효과적으로 진행된 것을 확인하는 계기가 되었다. 따라서 본 자료 분석방법은 앞으로 남극 페브리-페로 자료의 분석방법으로 사용될 것이며 속도가 빠른 워크스테이션에서 처리할 수 있도록 수정할 계획이다.

참고문헌

Bevington, P.R., Data reduction and error analysis for the physical sciences, McGraw-Hill, New York, 1969

Marquardt, D.W., An algorithm for least-squares estimation of nonlinear parameters, J. Soc. Ind. Appl., Math., vol.11, no.2, 431-441, 1963

Killeen, T.L., and P.B. Hays, Doppler line profile analysis for a multichannel Fabry-Perot interferometer, Applied Optics, 23, 612-620, 1983

```
c     PROGRAM REDFIT
c
c     This program performs least-squares fitting of the FPI data
c     using the linearization of function.
c
c     Input:
c      unit 18 -
c                  ddd = day number (1-366)
c      unit 19 - Instrument FPI Fourier coefficient file THUdddc.Dyy
c              where ddd = day number of PSCAN file
c                    c = colour (R=red, G=green, N=neon)
c                   yy = year of PSCAN file
c
c     Output:
c     unit 22 - data file SUMTHUdddc.Dyy
c     unit 21 - dark count file DARTHUdddc.Dyy
c     unit 20 - system file SYSTHUdddc.Dyy
c     where ddd, c, and yy have the meanings above
c     Note that files 21 and 20 are not usually kept.
c
      BYTE IPLOT,IFORM,IJUNK,IHOT
      COMMON /FRINGE/FPCNT(12), WEIGHT(12), TIME, DTIME,
      KWAVE,XPEAK,PMIN,KDATE(3),KREC,KKFP,FPAZ,FPZE,ITERS
      COMMON /SYSPAR/T1,T2,T3,T4,T5,T6,T7,HRPRES,JCAL
      COMMON /CALIBR/DARK(12),DARKI(12),KDARK
      COMMON /GAUS/AA(4), DELT(4), FPFIT(12), CHISQR, FLAMDA,
     + IFLAG
      COMMON /CONST/NCOEFS, NTERM, PCHFSR, TGAMMA, PI, VFSR
      COMMON /SHOW/CHAN(12)
      COMMON /STORE/CAL(17),DNORTH(17),DSOUTH(17),
     + DEAST(17), DWEST(17), ZENITH(17), DTHULE(17), DXTRA(17)
c
c     note: TGAMMA and VFSR for 1.116 cm gap and OI(6300A) wavelength
c
      DATA TGAMMA/1.4404E-5/, VFSR/8468./, NTERM/4/       ! (1.116 cm)
      DATA CHAN/1.,2.,3.,4.,5.,6.,7.,8.,9.,10.,11.,12./
c
c     read one record to get KDATE to compute NDAY
c
      PI=4.*ATAN(1.)
      KDARK=0                          ! set up dark counter
      DO 90 I = 1, 12
90    DARK(I) = 0.
      ACCEPT 99, IJUNK
      IF (IJUNK .EQ. 'Y') READ(18,99) JUNK          ! To get by <LF>
      ACCEPT 99, IHOT                               ! Hot channel
99    FORMAT(A1)
      CALL RDCOEF
```

```
      CALL INPUT(IPLOT,BG,SIG,GSWIND,GSTEMP,IFORM)
      CALL INIT
      IEND = 0                  ! read flag
      DO 100 II = 1, 5000       ! should not be ever this many fringes
      CALL READIN(II,IEND,KDAY,IFORM)
      IF (FPCNT(9) .EQ. 0.0) GOTO 100 ! bypass zero data
      IF (JCAL .EQ. 1) GOTO 100  ! bypass 6328 fringe
      IF (JCAL .EQ. 2) GOTO 100  ! bypass dark count
      IF (KWAVE .EQ. 5577) GOTO 100 ! bypass green line data
      IF (KWAVE .EQ. 7320) GOTO 100 ! bypass O+ data
      IF (KWAVE .NE. 6300 .AND. KWAVE .NE. 6301) PAUSE 'Invalid
      wavelength'
      IF (IEND .EQ. 5) STOP 'End of data analysis'
c
      CALL SETUP(BG,SIG,GSWIND,GSTEMP,JCAL,KDAY,IHOT)
      CALL FITTER(FPCNT,DTIME,ITERS,IPLOT)
      TYPE 260,KREC,TIME,ITERS,AA(1)/DTIME,AA(2)/DTIME,
     |  ,AA(3)*VFSR,DELT(3)*VFSR,AA(4),DELT(4),CHISQR,DTIME,KDAY
260   FORMAT(I4,F8.3,I3,'  ',2F7.2,' ',2(F7.0,F4.0),'  ',F8.2,F6.0,I4)
      CALL RECORD(KDAY)              ! store values in data array
c
100   ENDDO
c
      STOP        'End of data analysis'
      END
c
c-------------------------------------------------------------------------------
c
      SUBROUTINE INPUT(IPLOT,BG,SIG,GSWIND,GSTEMP,IFORM)
c
c     Subroutine to bring input information for program
c
      COMMON /CONST/NCOEFS, NTERM, PCHFSR, TGAMMA, PI, VFSR
      BYTE IPLOT,IFORM
      ACCEPT 1,IFORM
      ACCEPT 1,IPLOT
1     FORMAT(A1)
      ACCEPT *,BG,SIG,WINCON,GSTEMP
      GSWIND = WINCON/VFSR
      RETURN
      END
c
c-------------------------------------------------------------------------------
c
      SUBROUTINE OUTPUT(JCAL)
c
c     This subroutine outputs results on unit 22.
c
      COMMON /STORE/CAL(17),DNORTH(17),DSOUTH(17),
     |  DEAST(17),DWEST(17),ZENITH(17),DTHULE(17),DXTRA(17)
c
```

```
      GOTO (1,2,3,4,5,6,7,8), JCAL
1     WRITE(22,100)JCAL,  (CAL(JJ),JJ=1,17)
      RETURN
2     WRITE(22,100)JCAL,(ZENITH(JJ),JJ=1,17)
      RETURN
3     WRITE(22,100)JCAL,(DNORTH(JJ),JJ=1,17)
      RETURN
4     WRITE(22,100)JCAL,(DSOUTH(JJ),JJ=1,17)
      RETURN
5     WRITE(22,100)JCAL, (DEAST(JJ),JJ=1,17)
      RETURN
6     WRITE(22,100)JCAL, (DWEST(JJ),JJ=1,17)
      RETURN
7     WRITE(22,100)JCAL, (DTHULE(JJ),JJ=1,17)
      RETURN
8     WRITE(22,100)JCAL, (DXTRA(JJ),JJ=1,17)
100   FORMAT(1X,I2,1X,F7.3,F10.3,F7.3,F10.3,F7.3,F7.0,F6.0,
     | F6.0,F5.0,F8.2,F9.3,2F9.1,F8.0,2F5.0,F6.0)
      RETURN
      END
c
c-------------------------------------------------------------------------------
c
      SUBROUTINE RECORD(KDAY)
c
c     record data in arrays and output files
c     first select data according to direction
c
      COMMON /FRINGE/FPCNT(12), WEIGHT(12), TIME, DTIME,
      KWAVE,PEAK,PMIN,KDATE(3),KREC,KKFP,FPAZ,FPZE,ITERS
      COMMON /GAUS/AA(4), DELT(4), FPFIT(12), CHISQR,FLAMDA,
     | IFLAG
      COMMON /STORE/CAL(17),DNORTH(17),DSOUTH(17),
     | DEAST(17),DWEST(17),ZENITH(17),DTHULE(17),DXTRA(17)
c
      IF(FPZE.GT.120.)THEN
            KSET = 1
            CALL STORE(CAL,AA,DELT,CHISQR,KSET,KDAY)
            CALL OUTPUT(KSET)
            RETURN
      ELSE IF(ABS(FPZE-5.).LT.10.)THEN
            KSET = 2
            CALL STORE(ZENITH,AA,DELT,CHISQR,KSET,KDAY)
            CALL OUTPUT(KSET)
            RETURN
      ELSE IF(FPZE.GT.40..AND.ABS(FPAZ).LT.15.)THEN
            KSET = 3
            CALL STORE(DNORTH,AA,DELT,CHISQR,KSET,KDAY)
            CALL OUTPUT(KSET)
            RETURN
      ELSE IF(FPZE.LT.-20.AND.ABS(FPAZ).LT.15.)THEN
```

```
            KSET = 4
            CALL STORE(DSOUTH,AA,DELT,CHISQR,KSET,KDAY)
            CALL OUTPUT(KSET)
            RETURN
      ELSE IF(FPZE.GT. 20.  .AND.  FPZE.LT.100..AND.
     |      FPAZ.GT.80.)THEN
            KSET = 5
            CALL STORE(DEAST,AA,DELT,CHISQR,KSET,KDAY)
            CALL OUTPUT(KSET)
            RETURN
      ELSE IF(FPZE.LT.-20..AND.FPAZ.GT.80.)THEN
            KSET = 6
            CALL STORE(DWEST,AA,DELT,CHISQR,KSET,KDAY)
            CALL OUTPUT(KSET)
            RETURN
      ELSE IF(ABS(FPZE-65.).LT.5..AND.ABS(FPAZ+22.).LT.5)  THEN
            KSET = 7
            CALL STORE(DTHULE,AA,DELT,CHISQR,KSET,KDAY)
            CALL OUTPUT(KSET)
            RETURN
      ELSE
            KSET = 8
            CALL STORE(DXTRA,AA,DELT,CHISQR,KSET,KDAY)
            CALL OUTPUT(KSET)
            RETURN
      ENDIF
      RETURN
      END
c
c-----------------------------------------------------------------
c
      SUBROUTINE READIN(II,IEND,KDAY,IFORM)
c
c     Subroutine reads in the data from the input file.
c     Also collects dark counts when appropriate.
c     Modified to accept several types of formatted input simply by
c     editing the .RUN file.
c
      COMMON /FRINGE/FPCNT(12), WEIGHT(12), TIME, DTIME,
      KWAVE,PEAK,PMIN,KDATE(3),KREC,KKFP,FPAZ,FPZE,ITERS
      COMMON /SYSPAR/T1,T2,T3,T4,T5,T6,T7,HRPRES,JCAL
      COMMON /CALIBR/DARK(12),DARKI(12),KDARK
      BYTE IFORM
c
      IF (II .EQ. 1 .AND. IFORM .EQ. 'C') THEN
            READ(18, 41, END=999)  ILF, KREC, KKFP, FPAZ, FPZE, IPASS,
     + MODE, JOUT, T1, T2, T3, T4, T5, T6, WAVELN, KDATE, DTIME
      ELSE IF (II .EQ. 1 .AND. IFORM .EQ. 'D') THEN
            READ(18, 42, END=999)  ILF, KREC, KKFP, FPAZ, FPZE, IPASS,
     + MODE, JOUT, T1, T2, T3, T4, T5, T6, WAVELN, KDATE, DTIME
```

```
      ELSE IF (IFORM .EQ. 'A' .OR. IFORM .EQ. 'C' .OR. IFORM .EQ. 'E')
      THEN
            READ(18, 51, END=999)  KREC, KKFP, FPAZ, FPZE, IPASS,
     + MODE, JOUT, T1, T2, T3, T4, T5, T6, WAVELN, KDATE, DTIME
      ELSE IF (IFORM .EQ. 'B' .OR. IFORM .EQ. 'D' .OR. IFORM .EQ. 'F')
      THEN
            READ(18, 52, END=999)  KREC, KKFP, FPAZ, FPZE, IPASS,
     + MODE, JOUT, T1, T2, T3, T4, T5, T6, WAVELN, KDATE, DTIME
      ELSE IF (IFORM .EQ. 'G') THEN
            READ(18, 53, END=999)  KREC, KKFP, FPAZ, FPZE, IPASS,
     + MODE, JOUT, T1, T2, T3, T4, T5, T6, WAVELN, KDATE, DTIME
      ELSE IF (II .EQ. 1 .AND. IFORM .EQ. 'S') THEN
            READ(18, 61, END=999)  ILF, KREC, KKFP, FPAZ, FPZE, IPASS,
     + MODE, JOUT, T1, T2, T3, T4, T5, T6, WAVELN, KDATE, DTIME
      ELSE IF (II .EQ. 1 .AND. IFORM .EQ. 'V') THEN
            READ(18, 61, END=999)  ILF, KREC, KKFP, FPAZ, FPZE, IPASS,
     + MODE, JOUT, T1, T2, T3, T4, T5, T6, WAVELN, KDATE, DTIME
      ELSE IF (IFORM .EQ. 'S' .OR. IFORM .EQ. 'T' .OR.  IFORM.EQ.'X' .OR.
     + IFORM .EQ. 'U' .OR. IFORM .EQ. 'V' .OR. IFORM .EQ. 'W') THEN
            READ(18, 62, END=999)  KREC, KKFP, FPAZ, FPZE, IPASS,
     + MODE, JOUT, T1, T2, T3, T4, T5, T6, WAVELN, KDATE, DTIME
      ENDIF
c
      KDAY = MODE
      KWAVE = IFIX(WAVELN)
c
      IF (IFORM .EQ. 'A' .OR. IFORM .EQ. 'C') THEN
            READ(18, 100)  FPCNT, HRPRES, KSTEP, TIME1, JCAL, TIME2
      ELSE IF (IFORM .EQ. 'B' .OR. IFORM .EQ. 'D') THEN
            READ(18, 101)  FPCNT, HRPRES, KSTEP, TIME1, JCAL, TIME2
      ELSE IF (IFORM .EQ. 'E') THEN
            READ(18, 102)  FPCNT, HRPRES, KSTEP, TIME1, JCAL, TIME2
      ELSE IF (IFORM .EQ. 'F') THEN
            READ(18, 103)  FPCNT, HRPRES, KSTEP, TIME1, JCAL, TIME2
      ELSE IF (IFORM .EQ. 'G') THEN
            READ(18, 104)  FPCNT, HRPRES, KSTEP, TIME1, JCAL, TIME2
      ELSE IF (IFORM .EQ. 'S' .OR. IFORM .EQ. 'T') THEN
            READ(18, 110)  FPCNT, HRPRES, KSTEP, TIME1, JCAL
      ELSE IF (IFORM .EQ. 'U' .OR. IFORM .EQ. 'V') THEN
            READ(18, 120)  FPCNT, HRPRES, KSTEP, TIME1, JCAL
      ELSE IF (IFORM .EQ. 'W') THEN
            READ(18, 121)  FPCNT, HRPRES, KSTEP, TIME1, JCAL
      ELSE IF (IFORM .EQ. 'X') THEN
            READ(18, 122)  FPCNT, HRPRES, KSTEP, TIME1, JCAL
      ENDIF
c
      IF (FPCNT(9) .EQ. 0.0) RETURN                    ! zero data
c
      IF (IFORM .EQ. 'G'                               ! old Sondre data mods.
     .          .OR. IFORM .EQ. 'S' .OR. IFORM .EQ. 'X'
     .          .OR. IFORM .EQ. 'T' .OR. IFORM .EQ. 'U'
```

```
     .              .OR. IFORM .EQ. 'V' .OR. IFORM .EQ. 'W')
     .                    THEN                    ! specific Thule mods.
            TIME2 = TIME1
            DTIME = DTIME*MODE                    ! calculate int. period
            IF (JCAL .EQ. 1) THEN
                  SUM = 0.
                  DO 200 I = 1, 12
200               SUM = SUM + FPCNT(I)
                  SUM = SUM/12.
                  IF (SUM .LT. 10.) JCAL = 2  ! set dark flag
            ENDIF
      ENDIF
c
      IF (IFORM .EQ. 'F') THEN
            TIME2 = TIME1
      ENDIF
c
41    FORMAT(A1,2I4,2F7.1,3I4,6F7.2,F6.0,3I3,F9.1)        ! 'C' (line 1)
42    FORMAT(A1,2I4,2F6.1,3I4,6F7.2,F6.0,3I3,F7.1)        ! 'D' (line 1)
51    FORMAT(2I4,2F7.1,3I4,6F7.2,F6.0,3I3,F9.1)     ! 'A', 'C', 'E'
52    FORMAT(2I4,2F6.1,3I4,6F7.2,F6.0,3I3,F7.1)     ! 'B', 'D', 'F'
53    FORMAT(I5,I4,2F6.1,3I4,6F6.2,F6.0,3I3,F5.1)   ! 'G'
61    FORMAT(A1,2I4,2F6.1,3I4,6F6.2,F6.0,3I3,F5.1)        ! 'S', 'V' (line 1)
62    FORMAT(2I4,2F6.1,3I4,6F6.2,F6.0,3I3,F5.1)     ! 'S','T','X','U','V','W'
100   FORMAT(13F8.3,I5,F9.0,I2,F9.0)                ! 'A', 'C'
101   FORMAT(F7.3,11F8.3,F7.3,I5,F9.0,I4,F9.0)      ! 'B', 'D'
102   FORMAT(F7.3,12F8.3,I5,F9.0,I2,F9.0)           ! 'E'
103   FORMAT(12F8.3,F7.3,I5,F9.0,I4,F9.0)           ! 'F'
104   FORMAT(12F8.2,F7.3,I5,F9.0,I4,F8.2)           ! 'G'
110   FORMAT(F7.3,11F8.3,F7.3,I5,F9.0,I4)           ! 'S', 'T'
120   FORMAT(12F8.3,F7.3,I5,F9.0,I4)                ! 'U', 'V'
121   FORMAT(F9.3,11F8.3,F7.3,I5,F9.0,I2)           ! 'W'
122   FORMAT(12F8.2,F7.3,I5,F9.0,I4)                ! 'X'
c
      CALL TIMCON(TIME,TIME1,TIME2)
      WRITE(19,125)KREC,TIME,HRPRES,T1,T2,T3,T4,T5,T6,DTIME,JCAL,
     + KDAY
125   FORMAT(1X,I4,F9.3,F9.3,1X,6F7.2,F8.1,2I4)
      IF(JCAL.NE.2) RETURN
c
      SUM = 0.
      DO 130 KK=1,12
      DARKI(KK) = FPCNT(KK)
      SUM = SUM + DARKI(KK)
130   CONTINUE
      SUM = SUM/12.
      IF (IFORM .EQ. 'G'
     .              .OR. IFORM .EQ. 'S' .OR. IFORM .EQ. 'X'
     .              .OR. IFORM .EQ. 'T' .OR. IFORM .EQ. 'U'
     .              .OR. IFORM .EQ. 'V' .OR. IFORM .EQ. 'W')
     .                    THEN                    ! specific Thule mods.
```

```
            CALL DARKAV(KREC,TIME,KDAY)
        ELSE IF (SUM .LT. 1.5) THEN
            CALL DARKAV(KREC,TIME,KDAY)
        ENDIF
        RETURN
999     IEND = 5
        RETURN
        END
c
c-----------------------------------------------------------------
c
        SUBROUTINE TIMCON(TIME,TIME1,TIME2)
c
c       This subroutine determines the average TIME from the start time
c       TIME1 and the end time TIME2.
c
c       need to convert from HHMMSS to HH.fraction
c
        IH1 = TIME1/10000.
        IH2 = TIME2/10000.
        IMIN1 = (TIME1 - 10000.*FLOAT(IH1))/100.
        IMIN2 = (TIME2 - 10000.*FLOAT(IH2))/100.
        ISEC1 = TIME1 - 10000.*FLOAT(IH1)  - 100.*FLOAT(IMIN1)
        ISEC2 = TIME2 - 10000.*FLOAT(IH2)  - 100.*FLOAT(IMIN2)
        TIME1 = FLOAT(IH1) + FLOAT(IMIN1)/60.  + FLOAT(ISEC1)/3600.
        TIME2 = FLOAT(IH2) + FLOAT(IMIN2)/60.  + FLOAT(ISEC2)/3600.
        TIME = (TIME1 + TIME2)/2.
        RETURN
        END
c
c-----------------------------------------------------------------
c
        SUBROUTINE STORE(ARRAY,AA,DELT,ERRSUM,KSET,KDAY)
c
c       This subroutine collects information from the
c       two arrays AA and DELT and TIME and deposit it
c       into the appropriate array after conversion into
c       m/s and units/sec IP.
c
        COMMON /FRINGE/FPCNT(12), WEIGHT(12), TIME, DTIME,
        KWAVE,PEAK,PMIN,KDATE(3),KREC,KKFP,FPAZ,FPZE,ITERS
        COMMON /SYSPAR/T1,T2,T3,T4,T5,T6,T7,HRPRES,JCAL
        COMMON /CONST/NCOEFS, NTERM, PCHFSR, TGAMMA, PI, VFSR
        DIMENSION ARRAY(17),AA(4),DELT(4)
c
        IF(DTIME.EQ.0.)DTIME  = 500.
        ARRAY(1) = TIME                 ! time
        ARRAY(2) = AA(1)/DTIME          ! background
        ARRAY(3) = DELT(1)/DTIME        ! background  error
        ARRAY(4) = AA(2)/DTIME          ! signal
        ARRAY(5) = DELT(2)/DTIME        ! signal error
```

```
      ARRAY(6) = AA(3)  *VFSR          ! wind
      ARRAY(7) = DELT(3)*VFSR                ! wind error
      ARRAY(8) = AA(4)           ! temperature
      ARRAY(9) = DELT(4)         ! temperature error
      ARRAY(10) = ERRSUM               ! chi square error
      ARRAY(11) = HRPRES         ! etalon pressure
      ARRAY(12) = FPAZ           ! mirror azimuth
      ARRAY(13) = FPZE           ! mirror zenith
      ARRAY(14) = DTIME          ! singal integration period
      ARRAY(15) = JCAL           ! 0 = data, 1 = cal, 2 = dark
      ARRAY(16) = ITERS          ! number of iterations thru fit
      ARRAY(17) = KDAY           ! day number
      RETURN
      END
c
c---------------------------------------------------------------------
c
      SUBROUTINE DARKAV(KREC,TIME,KDAY)
c
c     determine running average for the dark count
c     assume a set of 5 exposures before starting the averaging.
c
      COMMON /CALIBR/DARK(12),DARKI(12),KDARK
      DATA DARK1/12*0./ DARK2/12*0./ DARK3/12*0./
      DATA DARK4/12*0./ DARK5/12*0./
      DIMENSION DARK1(12),DARK2(12),DARK3(12),DARK4(12),
     + DARK5(12)
c
      KDARK = KDARK + 1
      DO 10 KK=1,12
      DARK1(KK) = DARK2(KK)
      DARK2(KK) = DARK3(KK)
      DARK3(KK) = DARK4(KK)
      DARK4(KK) = DARK5(KK)
      DARK5(KK) = DARKI(KK)
10    ENDDO
c
      RDARK = FLOAT(KDARK)
      IF (KDARK .GT. 5) RDARK = 5.
      DO 20 KK = 1 , 12
      DARK(KK) = (DARK1(KK) + DARK2(KK) +
     + DARK3(KK) + DARK4(KK) + DARK5(KK))/RDARK
20    ENDDO
      WRITE(21,200)KREC,TIME,DARKI,KDAY
200   FORMAT(1X,I3,1X,F10.3,12F7.3,I4)
      RETURN
      END
c
c---------------------------------------------------------------------
c
      SUBROUTINE RDCOEF
```

```
c
c     This subroutine reads in the adjusted Fourier coefficients and
c     associated information
c
      COMMON /INSTRU/ AREA(12), COSCOE(20,12), SINCOE(20,12),
      PHI(12)
      COMMON /CONST/NCOEFS, NTERM, PCHFSR, TGAMMA, PI, VFSR
c
c     Read in the number of pairs of coefficients (NCOEFS), the free
c     spectral range (PCHFSR), the area under the channel 1 curve, which
c     was used for normalization (C1AREA), the reference pressure (REFPRS),
c     which is the pressure at the channel 1 peak, and then the phi values
c     (PHI) and adjusted coefficients themselves (AREA, COSCOE, and
      SINCOE)
c     - ignore some of the data, just read it into dummy variables
c
      READ(20,*) NCOEFS, NDAY,REFPRS,PCHFSR,C1AREA
      DO 200 II = 1, 12
            READ(20,*) N1, D1,AREA(II),PHI(II)
            PHI(II) = -PHI(II)          ! to convert from pres. to angle
      DO 200 JJ = 1, NCOEFS
            READ(20,*) D1,D2,D3,D4,COSCOE(JJ,II),SINCOE(JJ,II),D7
200   ENDDO

      RETURN
      END
c
c=====================================================
      SUBROUTINE INIT
c
c     This subroutine writes a header on the output summary file
c
      COMMON /CONST/NCOEFS, NTERM, PCHFSR, TGAMMA, PI, VFSR
c
      PWAVE = 6300.
      RMASS = 16.
      SPACE = 1.116
C1    SPACE = 0.62
      WRITE(22, 10) NCOEFS, PWAVE, RMASS,PCHFSR,SPACE, TGAMMA
10    FORMAT(1X,I5,2F6.0,2F8.4,F12.8)
      RETURN
      END

C=====================================================

      SUBROUTINE SETUP(BG,SIG,GSWIND,GSTEMP,JCAL,KDAY,IHOT)
c
c     prepare data and parameters for fitting
c
      COMMON /FRINGE/FPCNT(12), WEIGHT(12), TIME, DTIME,
      KWAVE,PEAK,PMIN,KDATE(3),KREC,KKFP,FPAZ,FPZE,ITERS
```

```
      COMMON /GAUS/AA(4), DELT(4), FPFIT(12), CHISQR,
     + FLAMDA,IFLAG
      COMMON /CALIBR/DARK(12),DARKI(12),KDARK
      COMMON /CONST/NCOEFS, NTERM, PCHFSR, TGAMMA, PI, VFSR
      LOGICAL*1 ILOOP
      BYTE IHOT
c
c     dark count correction
c     attempt to handle problem (low intensity) fringes
c
      do 90 kk = 1, 12
90    if (fpcnt(kk) .le. dark(kk)) goto 95
      go to 99
95    do 96 kk = 1, 12
96    fpcnt(kk) = fpcnt(kk)*dtime
      go to 70
99    DO 100 KK = 1, 12
      FPCNT(KK) = (FPCNT(KK) - DARK(KK)) * DTIME
c     IF (FPCNT(KK) .LE. 0. .AND. KK .NE. 9) PAUSE 'Signal <0 c/s'
100   CONTINUE
70    IMIN = 1
      IMAX = 1
      DO 80 KK=2,12
      IF(FPCNT(KK).LT.FPCNT(IMAX))GOTO 75
      IMAX = KK
75    IF(FPCNT(KK).GT.FPCNT(IMIN))GOTO 80
      IMIN = KK
80    CONTINUE
      PEAK = FPCNT(IMAX)
      PMIN = FPCNT(IMIN)
85    CONTINUE
c
c     First guesses
c
      AA(1) = BG
      AA(2) = SIG
      AA(3) = GSWIND
      AA(4) = GSTEMP
c
c     calculate initial fitting function, weights and chi square here
c
300   CONTINUE
      DO 320, I = 1, 12
      II = I        !avoid warning from Fortran compiler
320   FPFIT(I) = FUNCTN(II, AA)
c
      DO 400, II = 1, 12
      IF (ABS(FPCNT(II)) .LT. 1.E-10) THEN
            WEIGHT(II) = 1.
      ELSE
            WEIGHT(II) = 1. / ABS(FPCNT(II))
```

```
      ENDIF
400   CONTINUE
      IF (IHOT .EQ. 'Y' .OR. IHOT .EQ. 'y') THEN
c     WEIGHT(9) = 0.0
      WEIGHT(12) = 0.0
      ENDIF
      CHISQR = FCHISQ(FPFIT)
      IFLAG = 0
      RETURN
      END
C
C=================================================

      SUBROUTINE FITTER(FPCNT,DTIME,ITERS,IPLOT)
c
c     This routine performs the fitting by calling CURFIT repeatedly until
c     the chi sqares don't change much.
c
      BYTE IPLOT
      DIMENSION FPCNT(12)
      COMMON /GAUS/AA(4), DELT(4), FPFIT(12), CHISQR,FLAMDA,
      IFLAG
      COMMON /CONST/NCOEFS, NTERM, PCHFSR,TGAMMA, PI, VFSR
c
      TOLER = 1.0005
      FLAMDA = .0002
      ITERS = 0                !count the number of iterations
      OLDCHI = CHISQR          !store the old chi square
c
      DO 50 JJ = 1,15
      CALL CURFIT
      IF(DTIME.LE.0.)DTIME  = 1000.
      IF(IPLOT.EQ.'Y')TYPE
      5,ITERS,OLDCHI,CHISQR,AA(1)/DTIME,AA(2)/DTIME,
     + AA(3)*VFSR,AA(4)
5     FORMAT(I5,2F12.4,2F10.3,2F12.0)
      ITERS = JJ
      IF (CHISQR .GT. OLDCHI) THEN
            IF(IPLOT.EQ.'Y')CALL PROFIL(FPCNT,FPFIT,DTIME)
            RETURN  ! no improvement so quit
            ENDIF
      DIFF = OLDCHI/CHISQR - TOLER
      IF (DIFF .LT. 0.) THEN
            IF(IPLOT.EQ.'Y')CALL PROFIL(FPCNT,FPFIT,DTIME)
            RETURN  ! no improvement so quit
            ENDIF
      OLDCHI = CHISQR
50    ENDDO
      RETURN
      END
c
```

```
c-------------------------------------------------------------------------------
c
      SUBROUTINE PROFIL(FPCNT,FPFIT,DTIME)
c
c     This subroutine plots on the terminal screen the
c     calculated and source profiles for comparison.
c
      DIMENSION FPCNT(12),FPFIT(12)
      LOGICAL*1 BAR(24,40)
c
      IF(DTIME.LE.0.)DTIME  = 1000.
      DO 10 KK = 1,12
      FPCNT(KK) = FPCNT(KK)/DTIME
      FPFIT(KK) = FPFIT(KK)/DTIME
10    ENDDO
      PEAK = 0.
      FPEAK = 0.
      DO 20 KK = 1,12
      IF(PEAK.GT.FPCNT(KK))GOTO  20
      PEAK = FPCNT(KK)
20    ENDDO
      DO 30 KK = 1,12
      IF(FPEAK.GT.FPFIT(KK))GOTO  30
      FPEAK = FPFIT(KK)
30    ENDDO
c
      LL = 0
      LK = 0
      LJ = 0
35    LL = LL + 1
      LK = LK + 1
      NUMPT = IFIX(20. * FPCNT(LK)/PEAK + 0.5)
      DO 38 JJ = 1,NUMPT
      BAR(LL,JJ) = 1H*
38    ENDDO
      DO 39 JJ = NUMPT + 1, 40
      BAR(LL,JJ) = 1H
39    ENDDO
c
      LL = LL + 1
      LJ = LJ + 1
      NUMPT = IFIX(20. * FPFIT(LJ)/FPEAK + 0.5)
      DO 40 JJ = 1,NUMPT
      BAR(LL,JJ) = 1H+
40    ENDDO
      DO 42 JJ = NUMPT + 1, 20
      BAR(LL,JJ) = 1H
42    ENDDO
      IF(LL.LT.24)GOTO  35
c
      JJ = 21
```

```
55    JJ = JJ - 1
      TYPE 200,(BAR(LL,JJ), LL = 1, 24)
200   FORMAT(5X,'|',24A1)
      IF(JJ.GT.1)GOTO 55
      RETURN
      END
c
c=====================================================
c
      FUNCTION FUNCTN(NN, PARAM)
c
c     this subroutine calculates the value of the fitting function
c     at point NN using the coefficients in the array PARAM.
c
      DIMENSION  PARAM(4)
c
      COMMON /INSTRU/ AREA(12), COSCOE(20,12), SINCOE(20,12),
     +  PHI(12)
      COMMON /CONST/NCOEFS, NTERM, PCHFSR, TGAMMA, PI, VFSR
c
      FACT1 = 2.0*PI
      ARG1 = -(PI**2)*TGAMMA
      ARG2 = PARAM(3) + PHI(NN)
      RJ1 = AREA(NN)
      RJ2 = AREA(NN)
      SUM = 0.0
c
      DO 10 II = 1, NCOEFS
            FACT2 = FACT1*FLOAT(II)
            ARG3 = ARG2*FACT2
            ARG4 = ARG1*(FLOAT(II)*FLOAT(II))*PARAM(4)
            TCOS = COSCOE(II,NN)*COS(ARG3)
            TSIN = SINCOE(II,NN)*SIN(ARG3)
            TEXP = EXP(ARG4)
            SUM = SUM+(TCOS+TSIN)*TEXP
10    CONTINUE
c
      FUNCTN = PARAM(1)*RJ1  + PARAM(2)*(RJ2+SUM)
      RETURN
      END
C
C=====================================================
C
      SUBROUTINE FDERIV(NN, DQDA)
c
c     This subroutine calculates the derivatives for the fitting
c     function  we are using.
c
      DIMENSION DQDA(4)
c     COMMON /GAUS/AA(4), DELT(4), FPFIT(12), CHISQR, FLAMDA,
     +  IFLAG
```

```
      COMMON /INSTRU/AREA(12), COSCOE(20,12), SINCOE(20,12),
     + PHI(12)
      COMMON /CONST/NCOEFS, NTERM, PCHFSR, TGAMMA, PI, VFSR
c
      FACT1 = 2.0*PI
      ARG1 = -(PI**2)*TGAMMA
      ARG2 = AA(3) + PHI(NN)
      RJ1 = AREA(NN)            ! use laser set for now
      RJ2 = AREA(NN)            ! laser set of normalization constants
      SUM2 = 0.0
      SUM3 = 0.0
      SUM4 = 0.0
c
      DO 10 II = 1, NCOEFS
       FACT2 = FACT1*FLOAT(II)
       ARG3 = ARG2*FACT2
       ARG4 = ARG1*(FLOAT(II)**2)*AA(4)
       TCOS = COSCOE(II,NN)*COS(ARG3)
       TSIN = SINCOE(II,NN)*SIN(ARG3)
       TCOSR = COSCOE(II,NN)*SIN(ARG3)
       TSINR = SINCOE(II,NN)*COS(ARG3)
       TEXP = EXP(ARG4)
       SUM2 = SUM2 + (TCOS+TSIN)*TEXP
       SUM3 = SUM3 + ((-TCOSR*FACT2)+(TSINR*FACT2))*TEXP
       SUM4 = SUM4 + (TCOS+TSIN)*TEXP*(ARG4/AA(4))
10    CONTINUE
c
      DQDA(1) = RJ1
      DQDA(2) = RJ2 + SUM2
      DQDA(3) = AA(2)*SUM3
      DQDA(4) = AA(2)*SUM4
      RETURN
      END
c
c=============================================================
c
      FUNCTION FCHISQ(YFIT)
c
c     The function value is the reduced Chi-square:  FCHISQ = SUM
c     (WEIGHT*(FPCNT-YFIT)**2)/NFREE
c
      DIMENSION YFIT(12)
      COMMON /FRINGE/FPCNT(12), WEIGHT(12), TIME, DTIME,
      KWAVE,PEAK, PMIN,KDATE(3),KREC,KKFP,FPAZ,FPZE,ITERS
c
      CHISQ = 0.0
c
      DO 300 II = 1, 12
300   CHISQ = CHISQ + WEIGHT(II) * (FPCNT(II) - YFIT(II))**2
c
c     now make that the reduced Chi Square
```

```
c
      FCHISQ = CHISQ / 8.   !8 is the degree of freedom
      IF (IHOT .EQ. 'Y' .OR. IHOT .EQ. 'y') THEN
      FCHISQ = CHISQ / 7.   !7 is the degree of freedom  (85/86)
      ENDIF                 !for 11 channel
      RETURN
      END
C
C=============================================================
C
      SUBROUTINE CURFIT
c
c     This subroutine is described in Bevington' Book, "Data reduction
c     and Error analysis for the Physical Sciences" on page 237.
c     It willfind the least-squares fit to an arbitrary non-linear
c     function by linearizing the function.
c
c     PARAMETERS:
c     NTERM     number of parameters
c     AA   array of parameters
c     DELT      array of stnd. deviations for parameters AA
c     FLAMDA    factor for determining how much this will act like a gradient
c           search
c     FPFIT     array of calculated values for FPCNT
c     CHISQR    reduced chi square for the fit
c     IFLAG     flag indicates how fitting fails when it is not zero
c
c     SUBROUTINES NEEDED:
c     FUNCTN    (II, A) evaluates the fitting function for the IIth point
c     FCHISQ    (YFIT) finds the chi square for the fit
c     FDERIV    (II, DERIV) evaluates the derivatives of the fitting function
c     MATINV    inverts a square matrix and finds its determinant
c
      DOUBLE PRECISION ARRAY
      DIMENSION ALPHA(4,4), BETA(4), DERIV(4), BB(4), YFIT(12)
c
      COMMON /FRINGE/FPCNT(12), WEIGHT(12), TIME, DTIME,
      KWAVE,PEAK,PMIN,KDATE(3),KREC,KKFP,FPAZ,FPZE,ITERS
      COMMON /GAUS/AA(4), DELT(4), FPFIT(12), CHISQR,FLAMDA,
     + IFLAG
      COMMON /CONST/NCOEFS, NTERM, PCHFSR,TGAMMA, PI, VFSR
      COMMON /MATRIX/ ARRAY(4,4), DET
c
c     fill alpha and beta matrices
c
      DO 30 JJ = 1, NTERM
       BETA(JJ) = 0.
       DO 30 KK = 1, JJ
30     ALPHA(JJ,KK) = 0.
      DO 50  II = 1, 12
       IOUT = II
```

```
c
c     evaluate derivatives for each point
c
      CALL FDERIV(IOUT, DERIV)
c
c     evaluate gradient sum for each sought after parameter
c
      DO 46 JJ = 1, NTERM
        BETA(JJ) = BETA(JJ) + WEIGHT(II)*(FPCNT(II)  - FPFIT(II))
     +              * DERIV(JJ)
c
c     calculate cross terms
c
      DO 46 KK = 1, JJ
46      ALPHA(JJ,KK) = ALPHA(JJ,KK) +
     +  WEIGHT(II)*DERIV(JJ)*DERIV(KK)
50    CONTINUE
c     make the matrix symmetric
      DO 53 JJ = 1, NTERM
       DO 53 KK = 1, JJ
53    ALPHA(KK,JJ) = ALPHA(JJ,KK)
c
c     invert the matrix for solving the equations
c     first normalize the elements so the diagonals are 1.0
c
71    CONTINUE
      DO 74 JJ = 1, NTERM
       DO 73 KK = 1, NTERM
         RTEMP = SQRT(ALPHA(JJ,JJ) *  ALPHA(KK,KK))
         IF (ABS(RTEMP) .GT. 1E-36)  GO TO 73
         CHISQR = 1E8
         IFLAG = 1
         RETURN
73       ARRAY(JJ,KK)= DBLE(ALPHA(JJ,KK)/ RTEMP)
74       ARRAY(JJ,JJ)= DBLE(1. + FLAMDA)
c
      CALL MATINV
      IF (DET .NE. 0.) GOTO 80
      IFLAG = 3
      RETURN
80    CONTINUE
      DO 84, JJ = 1, NTERM
       BB(JJ) = AA(JJ)
       DO 84, KK = 1, NTERM
84    BB(JJ) = BB(JJ) + BETA(KK) *  SNGL(ARRAY(JJ,KK))
     +              / SQRT(ALPHA(JJ,JJ) *  ALPHA(KK,KK))
c
c     if the second parameter gets negative, it means that the curve
c     is upside down, which never happens, so we force it to be positive,
c     because with it negative, everything goes down the tubes.  We also
c     change the fourth one, although it's not as important to the rest
```

```
c
      BB(2) = ABS(BB(2))
      BB(4) = ABS(BB(4))
c
c     calculate the new chi square; if it is bigger than the old,
c     increase FLAMDA to make it more like a gradient search and try
c     again
c
      DO 92, II = 1, 12
        IOUT = II
92    YFIT(II) = FUNCTN(IOUT, BB)
      CHISQ1 = FCHISQ(YFIT)
      IF (CHISQR - CHISQ1) 95, 101, 101
95    FLAMDA = 10.0 * FLAMDA
c
c     this will prevent us from winding up in an infinite loop if we
c     get a hopeless fit
c
      IF (FLAMDA .LT. 1.E5) GO TO 71
c     CHISQR = FLAMDA          !substitute dummy variables here
      IFLAG = 2
      RETURN
c
c     update FPFIT, AA and CHISQR and calculate uncertainties
c
101   DO 103 JJ = 1, NTERM
        AA(JJ) = BB(JJ)
103   DELT(JJ) = SQRT(SNGL(ARRAY(JJ,JJ))/ALPHA(JJ,JJ))
      DO 106 II = 1, 12
106   FPFIT(II) = YFIT(II)
      CHISQR = CHISQ1
      FLAMDA = FLAMDA / 10.0
      RETURN
      END
C
C=============================================================
C
      SUBROUTINE MATINV
c
c     this subroutine will invert a square matrix
c     It will also calculate and return the determinant
c
c     See Bevington, p. 302 for more information
c
c     PARAMETERS:
c     ARRAY     a square matrix which will be repalced by it's inverse
c     DET  determinant
c
      DOUBLE PRECISION ARRAY,AMAX, SAVE
      DIMENSION IK(4), JK(4)
      COMMON /MATRIX/ ARRAY(4,4), DET
```

```
c
      DET = 1.0
      DO 100 KK = 1, 4
      AMAX = 0.0D0
21    DO 30 II = KK, 4
      DO 30 JJ = KK, 4
23    IF (DABS(AMAX) - DABS(ARRAY(II,JJ))) 24, 24, 30
24    AMAX = ARRAY(II,JJ)
      IK (KK) = II
      JK (KK) = JJ
30    CONTINUE
c
c     interchange rows and columns to put AMAX in ARRAY(KK,KK)
c
31    IF (AMAX) 41, 32, 41
32    DET = 0.0
      RETURN
41    II = IK(KK)
      IF (II - KK) 21, 51, 43
43    DO 50 JJ = 1, 4
      SAVE = ARRAY(KK,JJ)
      ARRAY(KK,JJ)= ARRAY(II,JJ)
50    ARRAY(II,JJ)= -SAVE
51    JJ = JK(KK)
      IF (JJ-KK) 21, 61, 53
53    DO 60 II = 1, 4
      SAVE = ARRAY(II,KK)
      ARRAY(II,KK)= ARRAY(II,JJ)
60    ARRAY(II,JJ) = -SAVE
c
c     accumulate elements of inverse matrix
c
61    DO 70 II= 1, 4
      IF (II - KK) 63,70, 63
63    ARRAY(II,KK)= - ARRAY(II,KK)/ AMAX
70    CONTINUE
71    DO 80 II = 1, 4
      DO 80 JJ = 1, 4
      IF (II-KK) 74, 80, 74
74    IF (JJ - KK) 75, 80, 75
75    ARRAY(II,JJ) = ARRAY(II,JJ)+ ARRAY(II,KK) * ARRAY(KK,JJ)
80    CONTINUE
81    DO 90 JJ = 1, 4
      IF (JJ - KK) 83, 90, 83
83    ARRAY(KK,JJ) = ARRAY(KK,JJ)/ AMAX
90    CONTINUE
      ARRAY(KK,KK) = 1.0D0 / AMAX
100   DET = DET * SNGL(AMAX)
c
c     restore matrix to former ordering
c
```

```
101   DO 130 LL = 1, 4
      KK = 5 - LL
      JJ = IK(KK)
      IF (JJ - KK) 111, 111, 105
105   DO 110 II = 1, 4
      SAVE = ARRAY(II,KK)
      ARRAY(II,KK)= - ARRAY(II,JJ)
110   ARRAY(II,JJ)= SAVE
111   II = JK(KK)
      IF (II - KK) 130, 130, 113
113   DO 120 JJ = 1, 4
      SAVE = ARRAY(KK,JJ)
      ARRAY(KK,JJ)= -ARRAY(II,JJ)
120   ARRAY(II,JJ) = SAVE
      CONTINUE
130   CONTINUE
      RETURN
      END
```