# UNCERTAINTY EVALUATION OF DELAYED NEUTRON DECAY PARAMETERS

A Dissertation

by

JINKAI WANG

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

December 2008

Major Subject: Nuclear Engineering

# UNCERTAINTY EVALUATION OF DELAYED NEUTRON DECAY PARAMETERS

A Dissertation

by

JINKAI WANG

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

| | |
|---|---|
| Chair of Committee, | Warren D. Reece |
| Committee Members, | Leslie A. Braby |
| | John R. Ford |
| | Michael A. Walker |
| Head of Department, | Raymond J. Juzaitis |

December 2008

Major Subject: Nuclear Engineering

# ABSTRACT

Uncertainty Evaluation of Delayed Neutron Decay Parameters. (December 2008)

Jinkai Wang, B.S., Shanghai Jiao Tong University

Chair of Advisory Committee: Dr. Warren D. Reece

In a nuclear reactor, delayed neutrons play a critical role in sustaining a controllable chain reaction. Delayed neutron's relative yields and decay constants are very important for modeling reactivity control and have been studied for decades. Researchers have tried different experimental and numerical methods to assess these delayed neutron parameters. The reported parameter values vary widely, much more than the small statistical errors reported with these parameters. Interestingly, the reported parameters fit their individual measurement data well in spite of these differences.

This dissertation focuses on evaluation of the errors and methods of delayed neutron relative yields and decay constants for thermal fission of U-235. Various numerical methods used to extract the delayed neutron parameter from the measured data, including Matrix Inverse, Levenberg-Marquardt, and Quasi-Newton methods, were studied extensively using simulated delayed neutron data. This simulated data was Poisson distributed around Keepin's theoretical data. The extraction methods produced totally different results for the same data set, and some of the above numerical methods could not even find solutions for some data sets. Further investigation found that ill-conditioned matrices in the objective function were the reason for the inconsistent results.

To find a reasonable solution with small variation, a regularization parameter was introduced using a numerical method called Ridge Regression. The results from the Ridge Regression method, in terms of goodness of fit to the data, were good and often better than the other methods. Due to the introduction of a regularization number in the algorithm, the fitted result contains a small additional bias, but this method can guarantee convergence no matter how large the coefficient matrix condition number. Both saturation and pulse modes were simulated to focus on different groups. Some of the factors that affect the solution stability were investigated including initial count rate, sample flight time, initial guess values.

Finally, because comparing reported delayed neutron parameters among different experiments is useless to determine if their data actually differs, methods are proposed that can be used to compare the delayed neutron data sets.

# DEDICATION

To those I love…

My wife

My parents

My sisters

And

….

# ACKNOWLEDGEMENTS

I would like to thank many people who helped me complete this project. Without their support, wisdom, guidance, and encouragement, this dissertation would have not been possible. First, I would like to give my special thanks to my committee chair, Dr. Reece, for his valuable technical advice, patience and continuous financial support for my study. It would have been impossible to finish this project without his kind help. I also want to thank my other committee members, Dr. Braby, Dr. Ford, and Dr. Walker, for their kind help and their time reviewing this dissertation.

Thanks are also given to my friends and colleagues and the department faculty and staff for their generous help. I want to extend my special gratitude to many people in the Nuclear Science Center, such as Alfred Hanna, Tom Fisher, and many others, for their support during my studies.

Finally, I want to thank my mother and father for their encouragement, and my wife for her patience and infinite love. I love you all, and no words that I could ever write would be adequate to express my thanks.

# NOMENCLATURE

| | |
|---|---|
| BFGS | Broyden-Fletcher-Goldfarb-Shanno |
| DN | Delayed Neutron |
| GCV | Generalized Cross-Validation |
| IMSL | International Mathematics and Statistics Library |
| LANL | Los Alamos National Lab |
| LM | Levenberg-Marquardt |
| MI-SVD | Matrix Inverse with SVD |
| MSE | Mean Square Error |
| QN | Quasi-Newton |
| RMSE | Root of Mean Square Error |
| RR | Ridge Regression |
| RSS | Residual of Sum Square |
| RSSR | Residual of Sum Square of Regression |
| RSSE | Residual of Sum Square of Error |
| SVD | Singular Value Decomposition |
| TSVD | Truncated Singular Value Decomposition |

# TABLE OF CONTENTS

# LIST OF FIGURES

Page

Page

# LIST OF TABLES

Page

Page

# 1. INTRODUCTION

When a fissionable nucleus undergoes neutron-induced fission, it emits fission neutrons and fission fragments. There are two types of fission neutrons: prompt neutrons and delayed neutrons. Prompt neutrons are released instantaneously (within 1E-13 seconds of the fission event). If the residual fission fragments are neutron rich, they will experience one or more successive beta decays to reduce their neutron excess. The more neutron-rich the fragments are, the more energetic and rapid the decay will be. If the excited energy of the daughter nucleus is higher than the binding energy of the last neutron, a neutron will be emitted as a mode of decay. These neutrons are called delayed neutrons, and the fission products that release them are called delayed neutron precursors. There are more than 271 known delayed neutron precursors and undoubtedly there are more unknown precursors with extremely low yields. A practical way to describe the emission rate of delayed neutrons is to divide the precursors into several groups according to their half-lives. The majority of precursors' half-lives vary from several hundred milliseconds to about one minute. The total number of delayed neutrons per fission depends on the fission material. For the sake of illustration, this project only considers U-235 fission. Typically, less than 1% of the fission neutrons per fission are delayed neutrons and 30% of them are emitted within 1.0 sec. Nonetheless, these delayed neutrons play a critical role in reactor time response.

This dissertation follows the style of Physical Review C.

If only prompt neutrons existed in a core, small positive reactivity changes would increase power rapidly and it would be extremely difficult to keep a reactor at steady power with a safe margin. However, delayed neutrons make the regulation of the chain reaction less difficult. Obviously, accurate information about the decay constants and the yields of delayed neutrons is important in reactivity control. Delayed neutrons are also important in the investigation of nuclear structure, the determination of the fissile content of fuel elements, and the development of economical fast reactor systems.

Great uncertainty in the measurement of delayed neutrons, especially of delayed neutrons from precursors with short half-lives, makes it difficult to obtain accurate information about the primary fission products. Most published papers focus on finding “best-fit” parameters for the grouped precursors and only a few discuss uncertainty estimation. Among these, the reported delayed neutron uncertainties are about ±4~5% for absolute yields, ±3~15% for group parameters, and ±10~20% for delayed spectra [1]. Various models and numerical methods have been used for decades to study the decay parameters. Their extracted parameters fit their measurement data very well in decay curves, although there is often a large discrepancy between the parameters reported by different investigators. No one has adequately explained why these excellent fits of the experimental data exhibit such large differences with reported decay parameters. An appropriate method is needed to improve optimization result and reduce uncertainty. The factors affecting instability should be investigated.

All delayed neutron experimental data contain error, often dominated by random error. The purpose of this research is to investigate the uncertainty of the delayed neutron relative yields and decay time constants. In particular, the numerical methods for extracting delayed neutron parameters are analyzed in detail to study the effect of this error on parameter estimation.

In this work, the “experimental data” is simulated using Monte Carlo methods. The simulated experimental data contains Poisson error but it is relatively precise due to the lack of measurement error or other variation that exist in real experiments. The only error in the generated data set is the variation based on the counts in a particular simulated channel. Delayed neutrons counts were simulated for both saturation and pulse irradiation mode so as to focus on precursors with long and short half-lives, respectively. Section 2 is a literature review of previous work in delayed neutron decay parameter estimation. Section 3 gives a basic introduction about delayed neutron emission and various equations. Section 4 highlights different numerical algorithms that are used to compute decay parameters and analyzes their fitted results. Section 5 introduces a new algorithm and compares its results with other algorithms. Section 6 presents the simulated results under different conditions. The experimental data sets are also used for comparison. Section 7 draws the conclusions based on the results. The calculated results and computer programs written in FORTRAN and MATLAB used for this research are included in Appendices A and B. The experimental data is listed in Appendix C.

# 2. LITERATURE REVIEW OF DELAYED NEUTRON PARAMETERS

For decades since their discovery, delayed neutrons have been studied because of their importance in nuclear reactor control. The present review focuses on the measurement of the delayed neutron emission rate and the parameters used to describe this rate (decay constants and relative yields) induced by fission neutrons with energies up to a few MeV.

The discovery of delayed neutrons was first reported by Roberts et al. [2] shortly after nuclear fission was confirmed in 1939. They thought the delayed neutrons were not photoneutrons caused by gamma radiation from the fission because the half-life of the neutron was different from the decay periods of the gamma radiation groups. Therefore, the delayed neutrons were unlikely to be related to the gamma radiation. Bohr and Wheeler [3] used a liquid drop nuclear model to explain the experiment of the delayed neutron emission. Their experiments proved that the delayed neutrons were released from highly excited beta decay daughters of the fission products.

Snell et al. [4] used a large slab of $U_3O_8$ embedded in a graphite block, which was irradiated by neutrons from $^9$Be (d, n) reaction in a cyclotron. Prompt and delayed neutrons were detected by a $BF_3$ detector in the graphite behind the $U_3O_8$. Some corrections were made due to the counting dead time and sample transfer time. But they ignored some effects such as scattering, and their final adjusted results were very different from the measured data.

Redman and Saxon [5] irradiated uranium and $^{239}$Pu samples in a graphite reactor and put the sample in a paraffin block containing a $BF_3$ detector, which was covered by lead. The absolute value for the ratio of $^{239}$Pu to $^{235}$U was obtained but no uncertainty was reported. Hughes et al. [6] performed the similar experiments. They used a stripping method to determine the half-lives and the yields of the delayed neutrons, i.e., they found the longest period group first, and then used this known value to get the next longest, and so on. They adjusted the irradiation time to emphasize the desired period as much as possible and found that five groups fit the measured data well. However, this method could only give an approximate solution because it dealt with the groups separately with different irradiation periods, and the counts from short half-lived groups were ignored because relatively short irradiation time resulted in large uncertainties.

Many improvements were subsequently made in delayed neutron research. A six-group exponential model was first proposed to describe the delayed neutron emission by Keepin et al. [7] [8] as described in the following expression:

$$C(t)=\sum_{i=1}^{6} A_i e^{-\lambda_i t} \tag{2-1}$$

C(t) is the count rate at time t; i is the number of groups with Group-1 having the longest half life; $A_i$ is the activity of Group i and $\lambda_i$ is the decay constant of Group i. They measured the periods, relative abundances, and absolute yields of delayed neutrons from fast and thermal fission of different nuclides. They used two approaches (prompt-burst irradiation and long-irradiation) to analyze the decay data at the Godiva Reactor in Los Alamos, USA. The sample transfer time was 50 ms. They assumed that the delayed neutron yields were energy independent. In this method, they empirically grouped delayed neutron precursors together based on the half-lives of precursors. A least-squares method was used to fit the data and they found that a six-group model was satisfactory to describe their data. The probable error of each parameter was determined by "weighted standard deviation of fit" and "inverse matrix from the solution of the weighted least-square equation" [8]. They found that the variation among group yields was much larger than the variation among group periods. In order to predict the most probable precursors for a given delayed neutron group, more than 100 possible delayed neutron precursors were studied by Keepin et al. [9]. They found that no satisfactory prediction was available for the sixth delayed neutron group. Later, additional experiments were performed to improve the precision of the parameters. The most widely used values were from another Keepin paper [10]. Keepin's delayed neutron decay parameters for $^{235}$U by fast and thermal neutrons are listed in Tables 2.1 and 2.2 below. At that time, Keepin used an iterative method to find the group parameters. These parameters are still widely accepted today. As shown later in Section 4.6, a large condition number of the matrix in this method can affect the calculated results significantly. The calculated errors shown in these tables were based on covariance matrix.

Table 2.1 Fast Fission Delayed Neutron Data from $^{235}$U (Keepin, 1957)

| Decay Constant (sec$^{-1}$) | Relative abundance |
|---|---|
| 0.0124±0.0003 | 0.038±0.003 |
| 0.0305±0.0010 | 0.213±0.005 |
| 0.111±0.004 | 0.188±0.016 |
| 0.301±0.011 | 0.407±0.007 |
| 1.14±0.15 | 0.128±0.008 |
| 3.01±0.29 | 0.026±0.003 |

Table 2.2 Thermal Fission Delayed Neutron Data from $^{235}$U (Keepin, 1965)

| Decay Constant (sec$^{-1}$) | Relative abundance |
|---|---|
| 0.0124±0.0003 | 0.033±0.003 |
| 0.0305±0.0010 | 0.219±0.009 |
| 0.111±0.004 | 0.196±0.022 |
| 0.301±0.011 | 0.395±0.011 |
| 1.14±0.15 | 0.115±0.009 |
| 3.01±0.29 | 0.042±0.008 |

In 1958, Cox et al. [11] reported measurements of delayed-neutron emission following the spontaneous fission of $^{252}$Cf using a $BF_3$ detector. The longest half-life of the delayed-neutron component was accurately determined by using essentially infinite collection times. The results were analyzed by a least squares method. They found that three groups, not six groups, were adequate to describe the delayed neutron emission. But they did not list the comparison nor show their own supporting evidence or data. In addition, they did not point out whether the three-group model was applicable to all or just some of the specified nuclides.

Masters et al. [12] used a modulated neutron source operated in antisynchronism with a long counter to measure the absolute and relative delayed-neutron yields of five fissionable isotopes for neutron-induced fission at 3.1 and 14.9 MeV. They compared the delayed-neutron yields at these two energies, and found that the delayed-neutron yields from 14.9-MeV fission averaged 40% less than those from fission by 3.1-MeV neutrons, in agreement with the prediction of Keepin. But the yields from 3.1-MeV neutrons were about 10% higher than those measured by Keepin et al. [9] from fast fission. They did not discuss the variances in half-lives of different groups, nor was there any discussion of the numerical method used to calculate their parameters.

Krick and Evans [13] studied the dependence of delayed neutron yields on the energy of the incident neutron. They pointed out that, for all fissionable isotopes studied, the delayed neutron yield was independent of the incident neutron energy in the range from 0.1 to 5 MeV. Krick and Evans measured, to an uncertainty of +/-10%, the absolute delayed neutron yields from fission of $^{233}$U, $^{235}$U, and $^{239}$Pu. Their conclusion is similar to that of Alexander and Krick's [14]. Alexander and Krick calculated the delayed neutron yields for $^{235}$U as a function of the incident neutron energy between thermal and 15 MeV based on fission data. They set the delayed neutron yields as constant and allowed the fission fractions to vary. Since experimental delayed neutron yields showed little variation with incident neutron energy below ~4 MeV, they thought the shape of the single-fission-model curve was constant from thermal to 5 MeV. The shape of delayed neutron yield as a function of incident neutron energy was identical with different amplitudes. Similar results were obtained by Caldwell et al. [15] from the irradiated sample of $^{232}$Th, $^{235}$U, $^{238}$U, and $^{239}$Pu. $^{3}$He proportional counters were embedded in polyethylene. They found that the delayed neutron yields did not change with the incident neutron energy.

Tuttle [16] used the empirical extrapolation method to evaluate the previous experimental data (such as delayed neutron yields, energy spectrum, group fractions and decay constants) for a number of fissionable materials with different incident neutron energies selected from thermal to 15 MeV. He recommended that the Keepin's data were better. Rudstam [17] derived the group parameters and spectra from the nuclear data and fission yields of the individual precursors using a six-group model. The calculated results were claimed to be consistent with the experimental data.

T. R. England and M. C. Brady et al. [18] [19] compared ENDF/B-V fission product yields (with six- and nine- group) to the result from CINDER-10 calculation. They found that for short periods (< 3 s) after the pulse, the ENDF/B-V data underestimates delayed neutron activity, and for long periods, it overestimates delayed neutron activity. A six-group representation fitted the data well. Increasing the group number to twelve did not significantly improve the fits. In their report, they gave little attention to the error analysis of these parameters.

R. W. Waldo, R. A. Karam, and R. A. Meyer [20] studied the time dependent thermal-induced gross delayed neutron yield for many nuclides from $^{232}$Th to $^{252}$Cf. The sample flight time was 300 msec. Twenty $^{3}$He ionization chambers were embedded in polyethylene. They developed an expression for the prediction of the absolute delayed neutron yield and the prediction of delayed neutron emission with time. The approach accurately predicted observed delayed neutron yields and delay characteristics. The results of this analysis were consistent with the experimental values. Waldo [21] used an inverse matrix iterative method to derive the delayed neutron parameters. Due to the long sample transfer time, only five groups were used. The long half-life group data agrees very well with Keepin's values. Not much information about error analysis was given.

G. Benedetti et al. [22] measured the delayed neutron emission from $^{233}$U, $^{237}$Np, $^{238}$Pu, $^{240}$Pu, $^{241}$Pu, and $^{241}$Am using a $BF_3$ counter surrounded by paraffin cylinder. A 1-cm thick lead shield was used to reduce gamma flux at the detector. Their instrumentation could not detect the sixth group of neutrons due to their short half-lives. The decay constants were derived from the analysis of the decay curves. They found that this method fit group yields very well. Compared to the results given by Tuttle [16], they found the delayed neutron yields from $^{240}$Pu and $^{241}$Pu were in good agreement, but the $^{233}$U yields had significant differences. Statistical error estimates were obtained by using an error propagation method. These errors were due to the uncertainty in the fission product yields and the neutron emission rates at the end of irradiation from the decay curves analysis. Systematic errors were not taken into account.

Y. Kaneko, F. Akino, and T. Yamane [23] evaluated the delayed neutron data for thermal fission of $^{235}$U. They used the least-square method to minimize the sum of the squared deviations between the measured values and the calculated values. After comparing the experimental results with the data from Keepin, ENDF/B-V and ENDF/B-IV, they found that the effective fraction, $\beta_{eff}$, was larger than Keepin's, but closer to ENDF/B-V data than ENDF/B-IV data. Decay constants values were close to Keepin's data for thermal fission.

Villani et al. [24] used the least squares method to decompose composite spectra measured at different delay time intervals following fission into group-wise delayed neutron spectra. They pointed out that the experimental uncertainties in the measured composite spectra were not the reason for unstable solutions (small changes in the measured spectra will lead to very different group spectra). They thought it was possible that the approximation of using the classical six-group fitting model was the cause of instability. But they did not investigate further.

H. H. Saleh et al. [25] measured delayed neutron yields and decay constants of $^{235}$U, $^{237}$Np, $^{241}$Am, and $^{243}$Am at the Texas A&M University TRIGA reactor. The sample flight time was 440 ms. By plotting neutron count rate versus time, the approximate yields and decay constants were found by a stripping method. These peeled values were then used as initial guesses in a weighted least squares fitting computer program that was derived specially to refine the initial estimates of the group yields and decay constants. Their calculated delayed neutron parameters were consistent with values recommended by Keepin [9], Tuttle [16], and Waldo [20], especially for $^{235}$U (see Table 2.3). Due to the long flight time (0.44 s), the shortest-lived group was ignored; and only five groups were fitted. Charlton et al. [26] [27] [28] used pulsed mode (60 ms irradiation time) in the same TRIGA reactor to catch the shortest-lived group of $^{237}$Np and $^{243}$Am providing new experimental results. Different irradiation times (180, 60, 20, and 5 s) were used to emphasize the different delayed neutron groups. The sample flight time was 0.5 sec and a seven-group fit was applied. Due to the small counts produced by pulsing technique, the long half life group data were obtained from previous experiments. Only statistical error was considered in error analysis.

Table 2.3 $^{235}$U and $^{237}$Np Group Constants at TRIGA Reactor

| Group | $^{235}$U | | $^{237}$Np | |
|---|---|---|---|---|
| | Decay Constant (sec$^{-1}$) | Relative Yield | Decay Constant (sec$^{-1}$) | Relative Yield |
| 1 | 0.0125±0.0009 | 0.036±0.006 | 0.0129±0.0006 | 0.040±0.002 |
| 2 | 0.036±0.002 | 0.239±0.039 | 0.0324±0.0010 | 0.233±0.017 |
| 3 | 0.111±0.007 | 0.195±0.033 | 0.1048±0.0019 | 0.19±0.01 |
| 4 | 0.300±0.005 | 0.390±0.065 | 0.341±0.013 | 0.322±0.027 |
| 5 | 1.100±0.025 | 0.111±0.018 | 0.85±0.06 | 0.193±0.007 |

D. Loaiza, G. Brunson, R. Sanchez, et al. [29] [30] studied the delayed neutron activity resulting from the fast induced fission of $^{235}$U and $^{237}$Np. “Instantaneous” and “infinite” irradiation were used. The sample transfer time was 0.110 s. The neutron detectors were 20 $^{3}$He tubes embedded in a polyethylene cylinder. A $^{3}$He proportional counter has high-efficiency, low dead time, and low gamma sensitivity. The absolute efficiency was 29.04% ± 0.6% determined by an Am/Li source. The system’s dead time was 0.46 ± 0.02 μs. The iterative Levenberg-Marquardt method was used to minimize the sum of squared differences between the experimental values and the fitted values. They compared the experimental results from a six-group model with the theoretical values, and they found that the measurements were in satisfactory agreement with

Keepin's data. The results are listed in Table 2.4. The decay of the delayed neutron activity for the saturation and instantaneous irradiations are represented by [30]:

$$N_S(t)=\sum_{i=1}^{m} A_i e^{-\lambda_i t} \qquad \text{(saturation irradiation)} \tag{2-2}$$

$$N_I(t)=\sum_{i=1}^{m} \lambda_i A_i e^{-\lambda_i t} \qquad \text{(instantaneous irradiation)} \tag{2-3}$$

Here, $N_S$ and $N_I$ are count rate from saturation and instantaneous modes separately, t is the independent variable time, $A_i$ is the abundance, and $\lambda_i$ is the decay constant of the i$^{th}$ group. The experimental data and the calculated results are depicted in Figure 2.1 (Obtained from [30]) for saturation and instantaneous irradiation. They had satisfactory agreement with Keepin's values.

Table 2.4 $^{235}$U and $^{237}$Np Group Constants in Godiva 4 in LANL

| Group | $^{235}$U | | $^{237}$Np | |
|---|---|---|---|---|
| | Decay Constant (sec$^{-1}$) | Relative Yield | Decay Constant (sec$^{-1}$) | Relative Yield |
| 1 | 0.0127±0.0001 | 0.0395±0.001 | 0.0123±0.0009 | 0.032±0.003 |
| 2 | 0.0315±0.0004 | 0.235±0.005 | 0.0284±0.0005 | 0.238±0.006 |
| 3 | 0.117±0.0064 | 0.207±0.008 | 0.0971±0.007 | 0.175±0.008 |
| 4 | 0.314±0.0107 | 0.381±0.011 | 0.296±0.014 | 0.360±0.017 |
| 5 | 1.37±0.0514 | 0.114±0.005 | 0.914±0.058 | 0.150±0.014 |
| 6 | 3.83±0.1138 | 0.0235±0.001 | 3.20±0.13 | 0.045±0.006 |

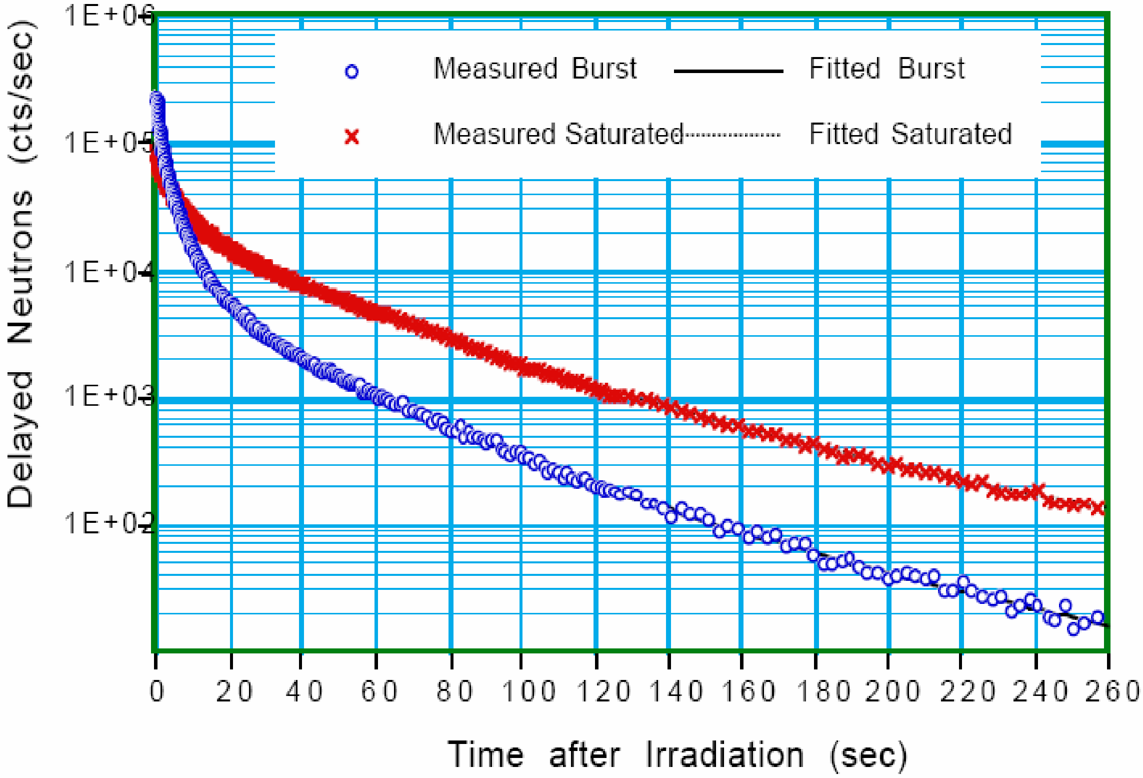


Figure 2.1 Measured and Fitted Delayed Neutron Decay Data for Saturated and Burst Irradiations [30]

J. Blachot, C. Chung and F. Storrer [31] used the summation calculation method to calculate the delayed neutron yields for 39 fissile systems using $P_n$ values and cumulative fission yields from JEF-2.2. They hoped that the new nuclear mass data for $^{235}$U thermal fission might improve the prediction of $P_n$ values, and that this calculation method might be a good way to provide a systematic estimation of the unmeasured yield values in terms of the atomic mass for each element with different neutron energies.

A new eight-group delayed neutron model based on a constant set of half-lives was proposed at Los Alamos in co-operation with IPPE at Obninsk by Spriggs et al. [32].

The researchers used the same set of eight half-lives for all fissionable systems. The half-lives of three longest-lived groups corresponded to the three dominant long-lived precursors: $^{87}$Br, $^{137}$I and $^{88}$Br. The main advantages of this model are: 1) a more consistent description of the delayed neutron emission from the longest-lived precursors to avoid distortions in reactivity measurement analysis (Keepin's six-group data did not accurately generate the asymptotic die-away time constants associated with the three longest-lived dominant precursors); 2) a single set of precursor half-lives for all fissile isotopes and incident neutron energies to calculate reactor kinetics without approximation (Keepin's six-group structure obtained different results for different isotopes and different incident neutron energies and causes approximations). According to the studies in Los Alamos, six-group and seven-group data were less good than an eight-group set in the analysis of strong negative reactivity insertion experiments [33].

G. D. Spriggs, J. M. Campbell [34] summarized the measured delayed neutron group parameters for 20 different fissionable isotopes. Two hundred and forty five sets of delayed neutron parameters were identified. The decay curves of each isotope were compared as a function of incident neutron energy. They found that the results from six-group model were still the best so far.

V. M. Piksaikin et al. [35] measured the absolute delayed neutron yield and group constants for fission of $^{235}$U, $^{239}$Pu, and $^{237}$Np induced by neutrons in a wide energy range from epithermal to 5 MeV at IPPE in Obninsk. Periodic irradiation with different irradiation times was used to focus on different delayed neutron groups. The pneumatic transfer system flight time was 150 ms. Thirty boron counters at three different distances were buried in polyethylene. Monte-Carlo correction factors were used to evaluate multiple scattering effects in the detection system. They used the successive approximation method [7] to optimize the generalized least squares fitting procedure. The Gauss-Newton method was used in the correction factor as the classical solution of the least squares method. The fitted data for six- and eight- group model were obtained for comparison. They noted that for the neutron energy range up to the threshold of the (n, n'f) reaction, the average half-life of delayed neutron precursors decreased when the energy of primary neutrons increases. They thought that the eight-Group model developed by Spriggs et al. [34] was a reliable procedure, but the improvement is not significant. The results from eight-group model are listed in Table 2.5. Piksaikin and Isaev [36] used an expression $v_d = aT^b$ to describe delayed neutron characteristics (a and b are empirical constants, T is the average half life of delayed neutron precursors, and $v_d$ is the total delayed neutron yield). These features were used to evaluate procedure that had a higher level of confidence than the evaluation based on a simple averaging procedure.

Table 2.5 An Eight-group Model Based on a Consistent Set of Half-lives [34]

| Group | Half-life (s) | Relative Yield: eight-group Fit | Relative Yield: Expanded eight-group Model |
|---|---|---|---|
| 1 | 55.7 | 0.030±0.10E-2 | 0.030±0.16E-2 |
| 2 | 24.5 | 0.172±0.40E-2 | 0.178±0.56E-2 |
| 3 | 16.3 | 0.103±0.30E-2 | 0.097±0.56E-2 |
| 4 | 5.21 | 0.163±0.40E-2 | 0.175±0.83E-2 |
| 5 | 2.37 | 0.364±0.60E-2 | 0.337±0.67E-2 |
| 6 | 1.04 | 0.018±0.10E-2 | 0.051±0.35E-2 |
| 7 | 0.424 | 0.128±0.60E-2 | 0.124±0.62E-2 |
| 8 | 0.195 | 0.022±0.10E-2 | 0.010±0.19E-3 |

S. B. Borzakov, et al. [37] measured the delayed neutron decay curves for the thermal-induced fission of $^{235}$U and $^{239}$Pu using pulse mode during the time interval from 5 to 730 ms after irradiation. They used a periodic irradiation technique to increase the number of precursors. The transfer time was about 1 sec, which resulted in larger uncertainties (15~35%) and variation in the fifth and sixth group parameters than those measured by Tuttle [16] and Waldo [20]. Compared to the results given by Brady and England [18] and Keepin [9] using six-, seven- and eight-group models, Borzakov, et al. [37] showed that a new seven-group model for this time interval was satisfactory. These comparitive results for $^{235}$U are shown in Figure 2.2 and the seven-group parameters are listed in Table 2.6. However, Wilson and England [38] came to a different conclusion. They used the CINDER-90 code to simulate the emission of over sixty delayed neutron precursors based on ENDF/B-VI basic nuclear data. The results from pulse and saturation irradiation equations were simulated to get total delayed neutron production. They found that using pulse irradiation fitted parameters to describe equilibrium irradiation results would increase deviations at large decay times; while using equilibrium irradiation results to fit the pulse irradiation results would result in greater deviation at small decay times. The maximum deviations were in the 2-5% range.

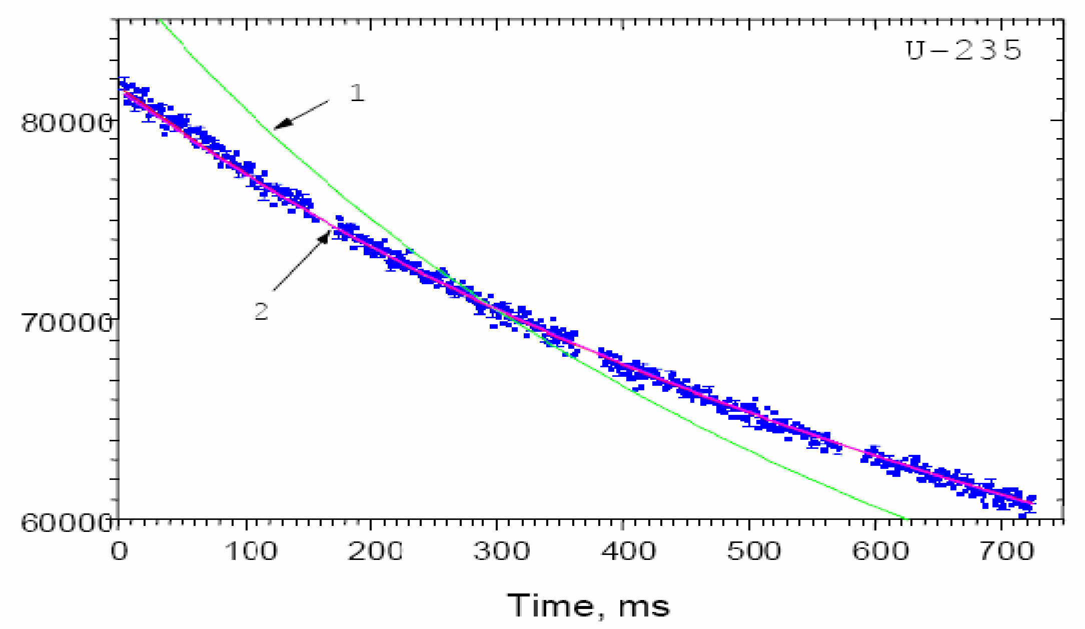
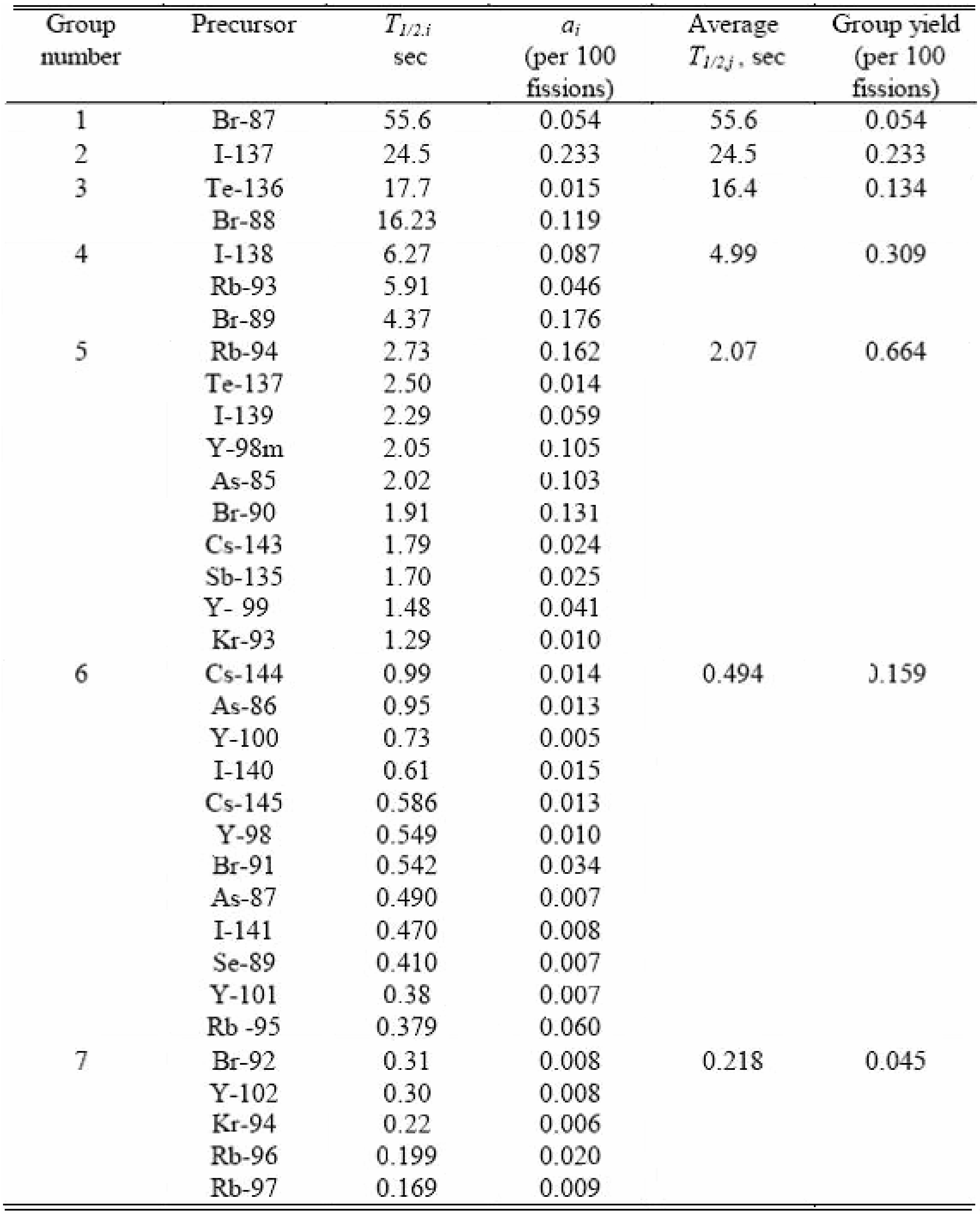


Figure 2.2 Experimental Data Obtained for $^{235}$U [37]. The curves were calculated using the following sets of delayed neutron parameters: Curve 1 - Brady, England [19]; Curve 2 - Keepin [7]. The curve calculated using the seven-Group model presented in this work coincides with Curve 2.

Table 2.6 Seven-group Parameters for Thermal Fission of $^{235}$U [37]



| Group number | Precursor | $T_{1/2,i}$ sec | $a_i$ (per 100 fissions) | Average $T_{1/2,j}$, sec | Group yield (per 100 fissions) |
|---|---|---|---|---|---|
| 1 | Br-87 | 55.6 | 0.054 | 55.6 | 0.054 |
| 2 | I-137 | 24.5 | 0.233 | 24.5 | 0.233 |
| 3 | Te-136 | 17.7 | 0.015 | 16.4 | 0.134 |
| | Br-88 | 16.23 | 0.119 | | |
| 4 | I-138 | 6.27 | 0.087 | 4.99 | 0.309 |
| | Rb-93 | 5.91 | 0.046 | | |
| | Br-89 | 4.37 | 0.176 | | |
| 5 | Rb-94 | 2.73 | 0.162 | 2.07 | 0.664 |
| | Te-137 | 2.50 | 0.014 | | |
| | I-139 | 2.29 | 0.059 | | |
| | Y-98m | 2.05 | 0.105 | | |
| | As-85 | 2.02 | 0.103 | | |
| | Br-90 | 1.91 | 0.131 | | |
| | Cs-143 | 1.79 | 0.024 | | |
| | Sb-135 | 1.70 | 0.025 | | |
| | Y- 99 | 1.48 | 0.041 | | |
| | Kr-93 | 1.29 | 0.010 | | |
| 6 | Cs-144 | 0.99 | 0.014 | 0.494 | 0.159 |
| | As-86 | 0.95 | 0.013 | | |
| | Y-100 | 0.73 | 0.005 | | |
| | I-140 | 0.61 | 0.015 | | |
| | Cs-145 | 0.586 | 0.013 | | |
| | Y-98 | 0.549 | 0.010 | | |
| | Br-91 | 0.542 | 0.034 | | |
| | As-87 | 0.490 | 0.007 | | |
| | I-141 | 0.470 | 0.008 | | |
| | Se-89 | 0.410 | 0.007 | | |
| | Y-101 | 0.38 | 0.007 | | |
| | Rb -95 | 0.379 | 0.060 | | |
| 7 | Br-92 | 0.31 | 0.008 | 0.218 | 0.045 |
| | Y-102 | 0.30 | 0.008 | | |
| | Kr-94 | 0.22 | 0.006 | | |
| | Rb-96 | 0.199 | 0.020 | | |
| | Rb-97 | 0.169 | 0.009 | | |

The Keepin's six-group fit and the eight-group fit proposed by Piksaikin [35] were calculated for comparison with some of the parameters (decay constants) fixed or variable. Borzakov et al. [37] found that the six-group fit with free abundances and decay constants provide satisfactory results with a small sum of squared residuals. Although the eight-group fit also gave a small sum of squared residuals, the difference was not significant. They concluded that it was not necessary to change the current six-group model.

B. Pfeiffer, K. L. Kratz, and P. Moller [39] listed the present status of the experimental delayed neutron precursor data, and compared with two model predictions: 1) an update of the empirical Kratz Herrmann formula (KHF), and 2) a unified macroscopic-microscopic models within the quasi-particle random-phase approximation (QRPA). The KHF results appeared to be more reliable than the QRPA results. And all half-life calculations agreed better with data for shorter half-lives. The mean deviation between calculated and experimental results was a factor of 2-5 for half-lives and 3-6 for $P_n$ values. Both of them depended on the model and half-life cutoff during calculation.

V. M. Piksaikin, L. E. Kazakov, et al. [40] [41], measured the relative yields and periods of individual groups of delayed neutrons (in $^{233}$U, $^{235}$U, $^{239}$Pu fission by epithermal neutrons and in $^{239}$Pu by primary neutrons 0.37-4.97 MeV) by periodic irradiation. They compared the experimental results with ENDF/BVI in terms of the average half-life of the precursor neutrons using the period- reactivity relation of the asymptotic period of the reactor. The group parameters were estimated using a least-square iteration method in a conventional six-group model. Within the uncertainty limits, the measured results differed substantially from the ENDF/BVI data. Piksaikin et al. [41] was the first to show the character and scale of the group parameters variation as a function of the excitation energy of the fissionable compound nucleus. For the primary-neutron energy range 2.85eV-5 MeV, the average half-life of precursor nuclei decreases by 10%.

S. G. Isaev et al. [42] studied the energy dependence of the relative delayed neutron yield related to individual precursors from fast fission of $^{237}$Np. They used periodic irradiation with sample flight time of 150 ms. The neutron detectors were an array of thirty boron counters located at three different distances along the concentric circles and buried in polyethylene. In data analysis, they found there was a strong correlation between the initial parameter inputs and the final calculated results of the delayed neutron yields with the half lives of 6.46 and 5.93 s; 2.3, 2.08 and 2.0 s; 1.002 and 0.86 s; 0.203 and 0.17 s. Therefore, they combined these precursors into four groups with half lives of 6.37, 2.09, 0.942, and 0.195 s. Together with another eight individual precursors, there are a total of twelve groups in their final results. The statistical errors were calculated in processing the experimental data by iterative least squares method. The systematic errors were estimated in the statistical analysis of the calculated relative yields, which were obtained by randomly changing the initial guessed inputs within ±40% of the results from the summation method. The correlation coefficients for the parameters showed a strong correlation between Group-3 and Group-4 in the six-group model [43]. Piksaikin et al. suggested combining these two groups and found that the new fifth group was in agreement with Keepin’s sixth group value.

D. Dore and J-C. David et al. [44] studied delayed neutron yields and spectra from photofission of actinides with bremsstrahlung photons below 20 MeV. One $^{3}$He detector was used. Three combinations of irradiation/counting (decay) times were applied: 140μs/30s, 5s/100s and 300s/300s. The absolute value of the DN yield for a long irradiation mode was 3.05 ± 0.20. An extraction method was used to get Group-1 and -2 decay parameters first in the long irradiation mode. Then, they subtracted these two groups and determined Group-3 and -4 parameters in middle irradiation mode. Finally, Group-5 and –6 parameters were found in short irradiation mode. The values for Group-6 were claimed to be very accurate with yield 0.085 ± 0.008 and half-life 0.174 ± 0.019 s. But for other groups, the values had large discrepancy compared to ENDF-B VI data. They only obtained the differing DN yield ratios because of the poor statistics in the experiment. Later, twelve $^{3}$He detectors were used to improve the total counts [45] and bremsstrahlung photons energies were between 12 and 18.5 MeV. The MCNPX modeling results were compared to experimental data. The Group-5 and Group-6 parameters were in good agreement. Only statistical error was considered based on several measurements, and no systematic uncertainty was evaluated.

V. M. Piksaikin et al. [46] studied the relative yield and half-life of their precursor nuclei from $^{238}$U fission by 14.2-17.9 MeV neutrons. Thirty SNM-11 counters (with $^{10}$B up to 80%) were located at three different concentric circles. Two irradiation times were used (180 and 300 sec). Sample flight time was 150 msec. The counting times were 524.5 and 724.5 sec. The dwell time was variable: 0.01 sec (150 channels), 0.02 sec (150), 0.1 sec (200), 1 sec (200), and 10 sec (30 or 50). They found that the average half-life of the precursor nuclei would increase if the primary neutron energy increases in that range. It was due to the opening of emissive nuclear fission channels. The systematic error from the detector blocking effect was also investigated. The least squares method was used for the calculation of the relative yields and half lives. The error they gave was the statistical error from several measurements.

A. Dos Santos et al. [47] used the multiple transient methods in their experiment at the IPEN/MB-01 research reactor to find the relative yield and decay constant of delayed neutrons. They used the least squares method to fit the parameters in a six-group model. This was done by iterative approach. First, the relative yields were fixed to find decay constants. Then, the process was inverted to find the relative yields. This was repeated until convergence was achieved. The only fixed parameter was Group-6 decay constants. Comparing the fitted parameters from the experiment to those from ENDF/B-VI revised library, they found that the uncertainties increased with the increase of the decay constants. They found that ENDF/B-IV.8 library overestimated the first decay constant. Their result was closer to JENDL-3.3 values. If all the decay constants were fixed, ENDF/B-VI.8 severely underestimated Group-1 relative yield with deviation as high as 26%, but the deviation from Group-6 was very small. If all the decay constants were fixed during the fit, the experimental values were severely different ENDF/B-VI, but they were in good agreement with the revised version of ENDF/B-VI (LANL version) library, especially with JENDL-3.3 library. But the multiple transient methods caused some difficulties in obtaining data from the short-lived precursor groups. Due to the extremely low power level (only 1 W), the detected counts were very low, and this increased error. They also did not compare the optimization differences with/without decay constants fixed.

B. Geslot et al. [48] investigated the influence of the delayed neutron group model on the fitted parameters. Only one hundred and forty precursors from JEFF-3.1 data library are used to get various sets of parameters for six, seven and eight groups, with six to fourteen variable parameters and in three incident neutron spectra (thermal, fast and monoenergetic 14 MeV). Constant time bin of 0.1 s was applied for 3000 points. The fitted results as well as their RMSE (root of mean squared error) from thirteen models (named from $J_1$ to $J_{13}$) are listed in Table 2.7. The comparison results between six-group data sets and Tuttle's data [16] are shown in Table 2.8. The highest uncertainties occurred in short decay groups and in the model with more free parameters. Small number of fitted parameters often gave high RMSE. Compared to the results from Tuttle (six-group) [16], Loaiza (seven-Group) [33], and Spriggs (eight-Group) [34], the fitted relative yields were significantly different. But the mean decay constants were very similar. The comparison for six-group was shown in Table 2.8. They concluded that the parameters were very dependent on the mathematical model and the experimental data, as well as the data analysis technique. No information was given about how to divide these precursors into six, seven, and eight groups. Only pulse mode was simulated. They also did not mention which numerical method was used to fit the parameters. Only a single fit from JEFF-3.1 data library was used for all parameters. Due to a constant dwell time 0.1 sec was used for all the data points, the fitted results have large correlation coefficients between short decay groups (few data points are available). The correlation coefficients between long decay groups are also very large because of the extremely low counts in the later counting period. The variation on the initial count rate in the model is not considered.

Table 2.7 Fitted Models Applied to Simulated Decay Curves [48]

| Spectrum | Group number | Parameter number | Fixed parameters | Data set name | RMSE | Max. Uncertainty (%) |
|---|---|---|---|---|---|---|
| Thermal | 6 | 12 | None | $J_1$ | 0.0205 | 45.5 ($\alpha_6$) |
| | | 10 | $\alpha_1, \lambda_1$ | $J_2$ | 0.0454 | 26.2 ($\alpha_6$) |
| | | 6 | $\lambda_i$ | $J_3$ | 0.23 | 2.7 ($\alpha_5$) |
| | 7 | 12 | $\alpha_1, \lambda_1$ | $J_4$ | 0.0113 | 63.2 ($\alpha_7$) |
| | | 7 | $\lambda_i$ | $J_5$ | 0.17 | 8 ($\alpha_4$) |
| | 8 | 14 | $\alpha_1, \lambda_1$ | $J_6$ | 0.0027 | 213 ($\alpha_8$) |
| | | 8 | $\lambda_i$ | $J_7$ | 0.087 | 8 ($\alpha_8$) |
| Fast | 6 | 12 | None | $J_8$ | 0.02 | 31.5 ($\alpha_6$) |
| | | 10 | $\alpha_1, \lambda_1$ | $J_9$ | 0.033 | 15.7 ($\alpha_6$) |
| | | 6 | $\lambda_i$ | $J_{10}$ | 0.23 | 2.7 ($\alpha_5$) |
| 14 MeV | 6 | 12 | None | $J_{11}$ | 0.031 | 31.8 ($\alpha_6$) |
| | | 10 | $\alpha_1, \lambda_1$ | $J_{12}$ | 0.033 | 18.8 ($\lambda_5$) |
| | | 6 | $\lambda_i$ | $J_{13}$ | 0.23 | 2.1 ($\alpha_5$) |

Table 2.8 Comparison between Six-group Data Sets [48]

| | | $J_1$ | $J_2$ | $J_3$ | Tuttle [16] |
|---|---|---|---|---|---|
| Relative Yield | 1 | 36.92 | 25.5 | 21.1 | 23.1 |
| | 2 | 155.4 | 154.8 | 148.7 | 153.3 |
| | 3 | 161.3 | 133.7 | 186.4 | 137.2 |
| | 4 | 255.2 | 279.6 | 260.7 | 276.5 |
| | 5 | 70.15 | 76.4 | 58.1 | 80.5 |
| | 6 | 21.03 | 30 | 24.5 | 29.4 |
| Decay Constant | 1 | 0.014 | 0.0124 | 0.0124 | 0.0124 |
| | 2 | 0.0334 | 0.0305 | 0.0283 | 0.0305 |
| | 3 | 0.137 | 0.113 | 0.119 | 0.111 |
| | 4 | 0.337 | 0.302 | 0.365 | 0.301 |
| | 5 | 1.25 | 1.03 | 1.29 | 1.14 |
| | 6 | 3.32 | 2.94 | 3.45 | 3.01 |

Note: $J_1$, $J_2$, and $J_3$ are defined in Table 2.7.

Different numerical methods based on different experimental data sets will generate different parameters. Further more, the fitted results are also very dependent on the data analysis methods as mentioned by Geslot [48]. The delayed neutron parameters are different among different researchers; some are significantly different. Several data libraries of delayed neutron parameters have been published. Table 2.9 lists delayed neutron decay parameters from different data libraries. These data are obtained from IAEA Nuclear Data Centre. It is obvious that these values differ significantly. JENDL-3.3 is the same as Keepin's [10] and Tuttle's [16] values. Most of them use six-group model, except for JEFF-3.1. The largest discrepancy occurs in short decay groups in all libraries.

Table 2.9 Thermal Neutrons Induced Delayed Neutron Fission Parameters in Different Data Libraries [49]

| | Group | ENDF/B-VII.0 (USA, 2006) | ENDF/B-VI.8 (USA, 2001) | JENDL-3.3 (Japan, 2002) | JEF-2.2 (Europe, 1992) | JEFF-3.1 (Europe, 2005) |
|---|---|---|---|---|---|---|
| Relative Yield | 1 | 0.0320 | 0.0350 | 0.0330 | 0.0350 | 0.0328 |
| | 2 | 0.1664 | 0.1807 | 0.2190 | 0.1807 | 0.1539 |
| | 3 | 0.1613 | 0.1725 | 0.1960 | 0.1725 | 0.0913 |
| | 4 | 0.4596 | 0.3868 | 0.3950 | 0.3868 | 0.1969 |
| | 5 | 0.1335 | 0.1586 | 0.1150 | 0.1586 | 0.3308 |
| | 6 | 0.0472 | 0.0664 | 0.0420 | 0.0664 | 0.0902 |
| | 7 | | | | | 0.0812 |
| | 8 | | | | | 0.0229 |
| Decay Constant | 1 | 0.0125 | 0.0133 | 0.0124 | 0.0127 | 0.0125 |
| | 2 | 0.0318 | 0.0327 | 0.0305 | 0.0317 | 0.0283 |
| | 3 | 0.1094 | 0.1208 | 0.1114 | 0.1160 | 0.0425 |
| | 4 | 0.3170 | 0.3028 | 0.3014 | 0.3110 | 0.1330 |
| | 5 | 1.3540 | 0.8495 | 1.1360 | 1.4000 | 0.2925 |
| | 6 | 8.6364 | 2.8530 | 3.0140 | 3.8700 | 0.6665 |
| | 7 | | | | | 1.6348 |
| | 8 | | | | | 3.5546 |

Studies to improve delayed neutron data have been carried out for many years. Great progress has been made for the individual precursors (or microscopic) level to get the delayed neutron yields, $P_n$ values, and half-lives. In the beginning, simple stripping techniques were used on the decay curve to get the delayed neutron parameters. With the improvement of detector efficiency, sensitivity and incident neutron energy independence, statistical variation has been greatly decreased during measurements, particularly with the application of multi-channel analyzers, and very satisfactory time resolution can be achieved. Later least squares fitting program codes were developed and analysis techniques were used, which avoid the subjective factors in graphical stripping methods and allow all the parameters to vary to fit the exponential equations. However, these parameters change greatly for different isotopes and incident neutron energies.

With the help of advances in computer simulation in advance, the experimental conditions can be optimized. Even given these advances, large discrepancies of the fitted parameters among different researchers still exist, promoting the investigation of parameter variation.

# 3. DELAYED NEUTRON EMISSION THEORY

After neutron-induced fission, the fission products usually undergo beta decay to reduce their neutron excess. However, in some cases, a daughter nucleus is formed after beta decay, such that the excitation energy is greater than the neutron-binding energy. This nucleus will directly emit a neutron as a mode of decay. This kind of neutron is called as delayed neutron. The delay time is governed by the half-life of the precursor nucleus.

A typical delayed neutron emission scheme is shown in Figure 3.1. $^{87}$Br is one of the dominant delayed neutron precursors from $^{235}$U fission.

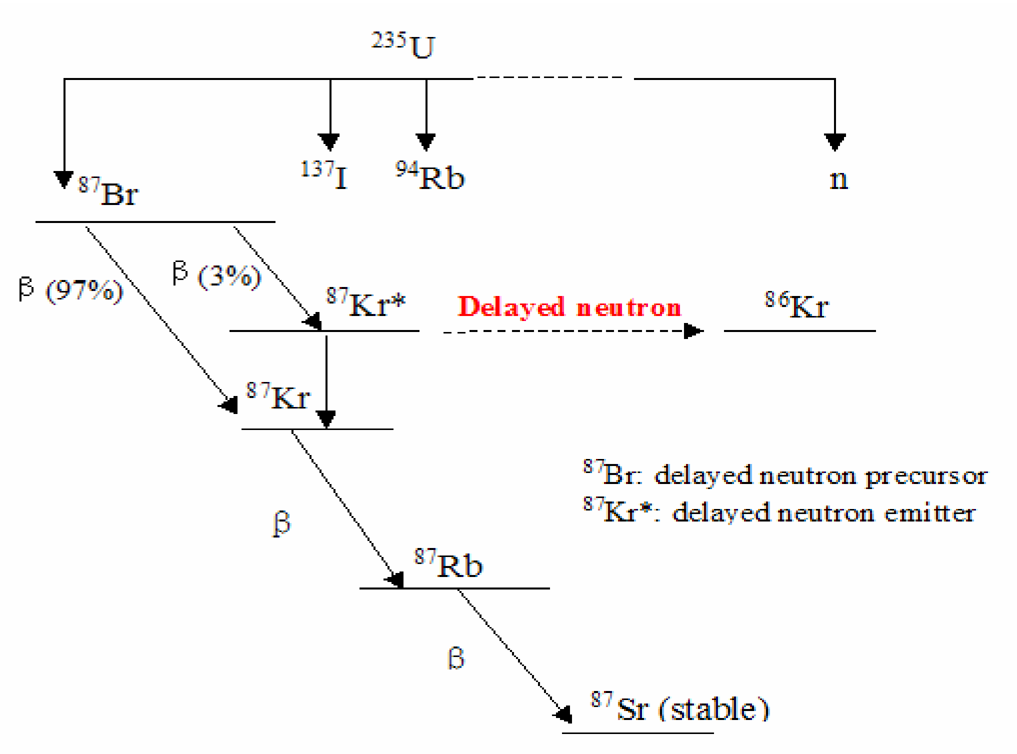


Figure 3.1 Delayed Neutron Emission Scheme [28]

Neutron generation time is defined as the time from the emission of one generation to the production of the next generation caused by the fission. It includes the period of being emitted, slowing down, and absorption to induce the next fission. In a fast reactor, the prompt neutron generation time is about 1E-6 seconds. In a thermal reactor, it is about 1E-4 seconds to 1E-5 seconds. Taking $^{235}$U fission as an example, the average time until emission of delayed neutrons from the precursors of $^{235}$U is about 12.5 seconds [50]. These delayed neutrons make a dominant contribution to the total neutron mean lifetime. The total delayed neutron yield is 0.0065, and taking the prompt neutron generation time as 5E-5 seconds, the weighted average generation time can be calculated as follows:

$$\begin{aligned}&Time_{prompt} * Yield_{prompt} + Time_{delayed} * Yield_{delayed}\\&= (5\times10^{-5}\,\mathrm{sec})(0.9935) + (12.5\,\mathrm{sec})(0.0065)\\&= 0.0813\,\mathrm{sec}\end{aligned}$$

Thus, mean generation time is increased significantly because of the delayed neutrons. This gives very important response time to regulate a reactor.

In this research, some of the delayed neutron decay properties will be explored mathematically. The number of precursor nuclei k denoted as $N_k(t)$ at irradiation time t in a sample can be expressed as:

$$\frac{dN_k(t)}{dt} = N\sigma_f\varphi\nu_k - \lambda_k N_k(t) \qquad (3\text{-}1)$$

Here, N is the number of actinide atoms, $\sigma_f$ the microscopic fission cross-section of the actinide nuclide, φ is the neutron flux, $\upsilon_k$ is the cumulative fission yield of the precursor k, and $\lambda_k$ is the decay constant of precursors. Therefore, we can get the number of the precursors during irradiation time $t_{irr}$:

$$N_k(t_{irr}) = N\sigma_f\varphi\nu_k\frac{1-e^{-\lambda_k t_{irr}}}{\lambda_k} \tag{3-2}$$

If the irradiation time is $t_{irr}$, after decay time t, the activity of the precursor k, $Act_k(t)$ is

$$Act_k(t) = N\sigma_f\varphi\nu_k(1-e^{-\lambda_k t_{irr}})e^{-\lambda_k t} \tag{3-3}$$

If the delayed neutron emission probability from precursor k is $P_k$, the total delayed neutron emission rate from precursor k at decay time t is

$$DN_k(t) = N\sigma_f\varphi\nu_k P_k(1-e^{-\lambda_k t_{irr}})e^{-\lambda_k t} \tag{3-4}$$

The total measured counts in the detector from all of the precursors during the counting time period $\Delta t = t_2 - t_1$ is given by

$$C = \varepsilon\int_{t_1}^{t_2}\sum_{k=1}^{n}[DN_k(t)]\,dt + B \tag{3-5}$$

Here, ε is the efficiency of the delayed neutron detection system; n is the total number of precursors; $t_1$ and $t_2$ are the beginning and ending counting time; and B is the background counts during the counting time period. The total measured count number during the counting time period $\Delta t = t_2 - t_1$ is:

$$C = N\sigma_f\varphi\varepsilon\sum_{k=1}^{n}[\frac{\nu_k P_k}{\lambda_k}(1-e^{-\lambda_k t_{irr}})e^{-\lambda_k t_1}(1-e^{-\lambda_k \Delta t})] + b\Delta t \tag{3-6}$$

Where $\lambda_k$ is the decay constant of precursor k, n is the total number of the precursors, and b is the background count rate.

There are more than 271 known delayed neutron precursors, and the production and decay for all of these individual precursors are mostly uncertain. Even if all these precursors were well known, a system of over 200 linked differential equations is also not of practical use. The widely accepted way to model delayed neutron production is to lump the precursors into several groups according to their half-lives. The purpose of this research is not to judge how many groups are better. A traditional six-group model will be used here. For a six-group model, the fraction of delayed neutrons and their main precursors are listed in the following Table 3.1 [10].

Table 3.1 Delayed Neutron Fractions for Different Nuclides

| Group | Possible precursor nuclei | Mean energy (MeV) | Average half-life of the precursor nuclei(s) | | | Delayed-neutron fraction (%) | | |
|---|---|---|---|---|---|---|---|---|
| | | | $^{235}$U | $^{239}$Pu | $^{233}$U | $^{235}$U | $^{239}$Pu | $^{233}$U |
| 1 | $^{87}$Br, $^{142}$Cs | 0.25 | 55.72 | 54.28 | 55 | 0.021 | 0.0072 | 0.0226 |
| 2 | $^{137}$I, $^{88}$Br | 0.56 | 22.72 | 23.04 | 20.57 | 0.14 | 0.0626 | 0.0786 |
| 3 | $^{138}$I, $^{89}$Br, $^{(93,94)}$Rb | 0.43 | 6.22 | 5.6 | 5 | 0.126 | 0.0444 | 0.0658 |
| 4 | $^{139}$I, $^{(93,94)}$Kr, $^{143}$Xe, $^{(90,92)}$Br | 0.62 | 2.3 | 2.13 | 2.13 | 0.252 | 0.0685 | 0.073 |
| 5 | $^{140}$I, $^{145}$Cs | 0.42 | 0.61 | 0.618 | 0.615 | 0.074 | 0.018 | 0.0135 |
| 6 | (Br, Rb, As etc) | - | 0.23 | 0.257 | 0.277 | 0.027 | 0.0093 | 0.0087 |
| | | Total | | | | 0.64 | 0.21 | 0.26 |

One of the most widely accepted models to predict delayed neutron production is Keepin's six-group model [10]. Some others used seven groups, such as Spriggs [32], Borzakov [37], or eight groups, such as Piksaikin [36], etc., these models are not widely accepted. The delayed neutron count rate CR(t) at decay time t after irradiation time $t_{irr}$ can be expressed as a summation of six exponential groups.

$$CR(t) = P\sum_{i=1}^{6} A_i e^{-\lambda_i t}(1 - e^{-\lambda_i t_{irr}}) + b \qquad (3\text{-}7)$$

Here, P is a proportionality constant depending on the reactor power level, the sample properties, detection efficiency, micro cross section of fissional nuclide. $A_i$ and $\lambda_i$ are the relative yield (the fractional abundance) and decay constant of group i, and b is the background count rate.

The total counts during time $\Delta t = t_2 - t_1$ can be derived as

$$\begin{aligned} C &= P\int_{t_1}^{t_2}\left(\sum_{i=1}^{6}[A_i(1-e^{-\lambda_i t_{irr}})e^{-\lambda_i t}]\right)dt + b\Delta t \\ &= P\sum_{i=1}^{6}[\frac{A_i}{\lambda_i}(1-e^{-\lambda_i t_{irr}})e^{-\lambda_i t_1}(1-e^{-\lambda_i \Delta t})] + b\Delta t \qquad (3\text{-}8) \\ &\approx P\sum_{i=1}^{6}[A_i(1-e^{-\lambda_i t_{irr}})e^{-\lambda_i (t_1+t_2)/2}]\Delta t + b\Delta t \end{aligned}$$

The above equation is a simplified form of Eq.(3-6). If $\Delta t$ is small enough, the integral part during time $\Delta t = t_2 - t_1$ can be simplified as the middle point value of the period multiplied by time period $\Delta t$ . This simplified expression is very helpful in computing the first and second derivatives of the unknown parameters. Even for a 5.0 sec dwell time, the calculated count difference is as low as 0.07% or less. Here, Eq.(3-8) can be further simplified under the following conditions:

1) For saturation irradiation, $(1-e^{-\lambda_i t_{irr}}) \approx 1$;

2) Background counts are ignored

The total counts during time period $\Delta t = t_2 - t_1$ can be derived from Eq. (3-8):

$$C = P \sum_{i=1}^{6} A_i e^{-\lambda_i (t_1 + t_2)/2} \Delta t \tag{3-9}$$

The delayed neutron count rate $CR_S(t)$ in saturation mode at time t is:

$$CR_s(t) = P \sum_{i=1}^{6} A_i e^{-\lambda_i t} \tag{3-10}$$

Actually, the above conditions can be easily achieved under current experimental conditions. The above equation is relatively more useful solution for the long half-life groups, such as Group 1 & 2. Because of the difficulties in capturing data from the short half-lived groups, the fitted results for Group-5 and 6 have large deviations [38]. The pulse irradiation method is used to focus on the short half-life groups, such as group 5 and 6 (see [29] and [38]). For a single pulse irradiation, the count rate at time t following fission is expressed as:

$$CR_p(t) = P \sum_{i=1}^{6} \lambda_i A_i e^{-\lambda_i t} \tag{3-11}$$

# 4. NUMERICAL METHODOLOGIES

Keepin et al. [9] were the first to propose the six-group model, that is, the delayed neutron activity as a function of time could be expressed by the sum of six exponentially decaying groups (see Eq.(3-10) and Eq.(3-11). The expressions in Eq.(3-10) and Eq.(3-11) are for saturation and pulse simulation respectively. The value of the sum of weighted squared residuals (marked as $E^2$, see below) between the measured count and the theoretical count from the model is used to judge the goodness of the fitted group parameters. There are thirteen parameters in the fitting equation: six relative yields and six decay constants, the last unknown parameter is a constant used to fit the initial count rate. The smaller the value of $E^2$, the better the fitted group parameters in the model matches the data. To find appropriate group parameters, the least squares method is widely used to minimize the sum of the weighted squared differences ($E^2$) between the measured values and the calculated results. If $y_i$ represents the $i^{th}$ observed count at time interval $\Delta t_i$, $\mathbf{x}$ is a vector of unknown parameters, and $t_i$ is the independent variable time after irradiation, objective function can be expressed as:

$$E^2=\sum_{i=1}^{m}\frac{1}{y_i}\left(y_i-f(\mathbf{x},t_i)\right)^2 \qquad [\mathbf{x}=(A_1,\ldots,A_n,\lambda_1,\ldots,\lambda_n)]$$

$$f(\mathbf{x},t_i)=\begin{cases} P\Delta t_i\sum_{j=1}^{n}A_j e^{-\lambda_j t_i} & \text{(saturation irradiation)} \\ P\Delta t_i\sum_{j=1}^{n}\lambda_j A_j e^{-\lambda_j t_i} & \text{(pulse irradiation)} \end{cases} \tag{4-1}$$

Here, f($\mathbf{x}$, $t_i$) is the fitted count at time $t_i$, $\Delta t_i$ is dwell time for $i^{th}$ observation; $A_j$ and $\lambda_j$ are the abundance and decay constant of group j, m and n are the number of the measurements and the number of group parameters respectively; P is a constant of proportionality depending on the reactor power level, the sample properties, detector efficiency, etc.

Since the above equation is a nonlinear least squares problem, an optimal solution usually cannot be found analytically. Iterative methods are used and initial estimations are used to start the iterations. Unreasonable initial values sometimes direct the algorithm away from a global minimum or even prevent convergence at all. There is no absolutely precise method for this problem. The methods used in the literature and described below seek to find an appropriate estimation while keeping the uncertainties as small as possible. Several numerical algorithms have been used, namely, the Matrix Inverse method, the Levenberg-Marquardt method, and the quasi-Newton method. The ordinary least squares method, the basis of the Matrix Inverse method, was widely used in previous research to solve this optimization problem.

To evaluate these methods, data sets were generated using Monte Carlo methods to simulate the experimental data as follows. Keepin's original parameters are used to generate simulated experimental data sets. The purpose of the simulation is to check whether the parameters extracted by the various algorithms are the same or close to the Keepin's initial values. If this is true, the simulated data sets from different random numbers should have stable solutions. Otherwise, the extracted values are unstable. If the objective optimization value $E^2$ cannot be guaranteed to be the smallest, the solution is not global and the solution only reaches a local optimization.

## 4.1 Data Generation

In this project, the experimental data are simulated by Monte Carlo methods. It is assumed that there are no experimental errors involving flight time error, detector efficiency, dead time, energy dependency, background noise and other sources of radiation. The only error source is from the statistic variation due to Poisson distribution. The simulated data sets are Poisson distributed along Keepin's theoretical values.

In saturation mode, the assumed count $y_{Keepin}(t_i)$ during time $\Delta t_i$ would be:

$$y_{Keepin}(t_i) = P\Delta t_i \sum_{j=1}^{n} A_j e^{-\lambda_j t_i} \tag{4-2}$$

Where i is the time channel number during the counting period, $\Delta t_i$ is the dwell time for the $i^{th}$ observation, $t_i$ is the mid-point of time interval $\Delta t_i$, and n is the number of groups assumed (n=6 in Keepin's model), $A_j$ and $\lambda_j$ are the "true" yield (the fractional abundance) and decay constant for group j, and P is a constant of proportionality depending on the reactor power level, the sample properties, detector efficiency, etc.

To generate the "experimental data", the simulated counts are Poisson distributed around the theoretical values calculated by Eq.(4-2) in each time channel. No other error is included. The only scatter about the mean is dictated by Poisson statistics. Poisson distribution is done using an inverse normal distribution. The $i^{th}$ "experimental count" $y(i)$ is simulated using following method:

$$\begin{aligned}&\mathrm{rand{=}RAN(seed)}\\&\mathrm{y(i){=}INT}\{\mathrm{y_{Keepin}(i){+}ANORIN(rand)}*\sqrt{\mathrm{y_{Keepin}(i)}}+0.5\}\end{aligned} \tag{4-3}$$

Here, "rand" is a random number generated by FORTRAN function RAN with an integer as a seed and "rand" varies between 0 and 1.0; $y_{Keepin}(i)$ is the theoretical count from Keepin's model in the $i^{th}$ time channel (see Eq.(3-10) and Eq.(3-11)); INT is a FORTRAN function to round off the number as an integer; function ANORIN is a function which is the inverse of the standard normal cumulative distribution from IMSL library. The data generation for pulse mode is similar as saturation mode except for the difference of theoretical function. The detailed information about data generation method is written in FORTRAN code in Appendix A.1, A.2 or A.3.

The counts in each dwell time (counting time channel) should be above 2,500, if possible, to ensure less than 2% relative deviation from the mean. If the time increment is too large, the counts change rapidly during each time increment and not enough information can be obtained for the shorter half-lived groups. If the dwell time is too short, the Poisson variation will be too big and the relative deviation will increase. Given these constraints, different time steps are chosen according to the count rate changes during decay period. During 0 to 10 sec, the dwell time is set to be 0.025 sec, during 10 to 100 sec, it is set to 0.5 sec, and during 100 to 300 sec, and it is set to 5.0 sec. The shortest dwell time 0.025 sec is selected based upon the initial count rate so that the shortest half-lived group information can be studied. If it is too small, a lot of channels are needed during counting period. This may be not applicable for some counting devices. Take MCA device (Model # MP2-2E, Canberra Corp., CT) in Nuclear Science

Center at Texas A&M University as an example, the total channels are 16024. If the total counting time is 300 sec, the time bin around 0.020 sec is appropriate. Considering the extra time to start counting in advance before the sample arrives, 0.025 sec time bin is a good selection. What's more, too few counts will also increase the statistical error. If the dwell time is too large, not enough information can be obtained for short half-lived groups. The 5.0 sec is selected in the later counting period because of the low count rate at that time and long half-life of Group-1 (around 55 seconds). Very long counting time is not necessary since the measurement error will increase greatly due to the rapid fall-off of the delayed-neutron emission when it is close to or below the experimental noise (see Figure 4.1). The following figure assumes that the initial count rate is 4.0E5 cps in both modes.

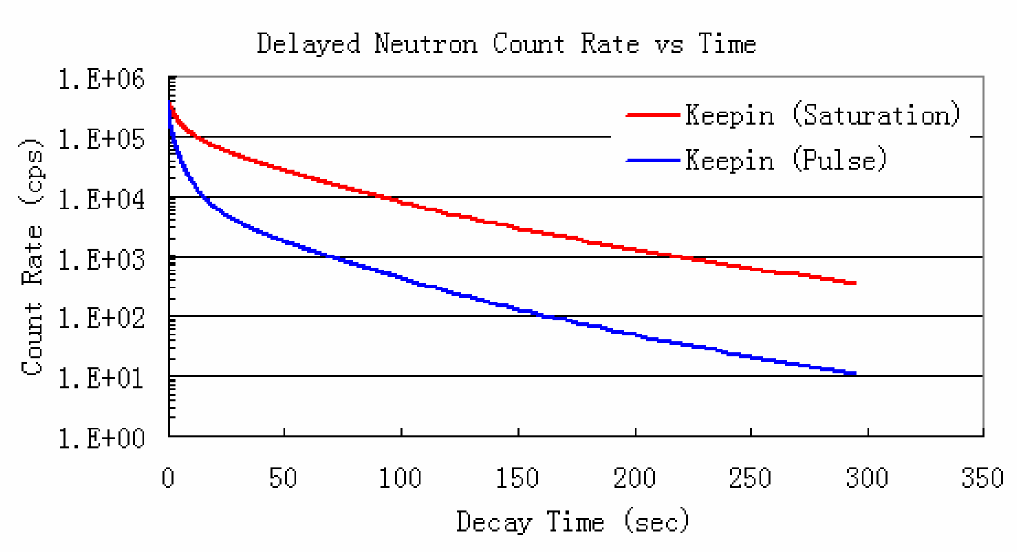


Figure 4.1 Delayed Neutron Count Rate with Time

As mentioned in Eq.(3-9), the theoretical count during the given dwell time is derived as the product of dwell time by the mid-point of the count rate during this time channel. This is explained in Figure 4.2.

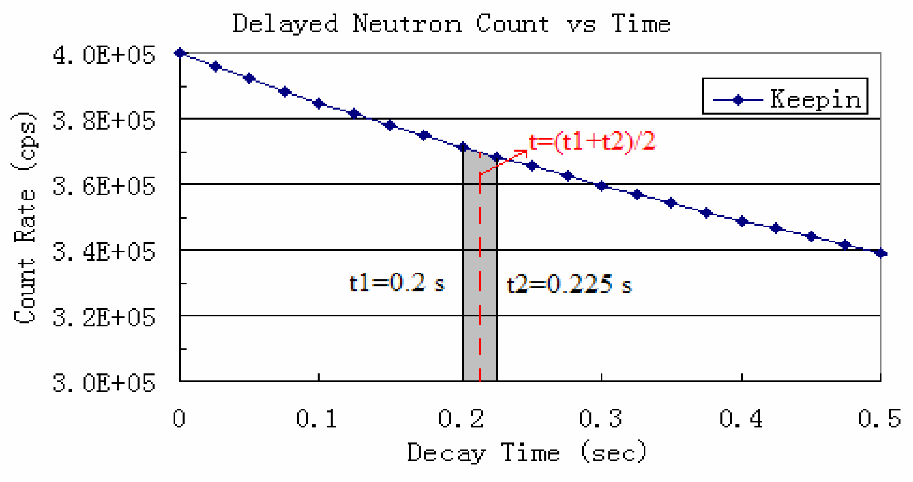


Figure 4.2 Illustration of Count Calculation in the Dwell Time

In Figure 4.2, the count during decay time 0.2 to 0.225 for the given dwell time 0.025 sec is simplified as the product of dwell time, 0.025 sec, by the count rate of the middle time, 0.2125 sec. This simplification is very helpful to reduce the complexity in computing the first and second derivatives of the unknown parameters. The theoretical count during dwell time should be derived from the integral of the count rate over the assigned time period. However, this simplified method results in little computational error. For very large dwell times, e.g., 5.0 sec, the relative error from this simplified method is 0.07% or less. This small error is negligible, but this method greatly reduces the computational complexity.

To make sure that the simulated data is not significantly different from Keepin's expected data, a Chi-square test is used to compare the difference at different decay time intervals. $\chi^2$ requires the sample data to be discrete and independent. The chi-square goodness-of-fit test is applied to binned data (i.e., data put into classes). It requires a sufficient sample size in order for the chi-square approximation to be valid. $\chi^2$ is a measure of the distance of the observed counts from the expected counts, and it is an one-tailed test. $\chi^2$ is defined as:

$$\text{Chi-square test: } \chi^2 = \sum \frac{(\text{observed - expected})^2}{\text{expected}} \tag{4-4}$$

Large values of $\chi^2$ are the evidence that the observed values are far from what we would expect if the hypothesis were true.

A statistical hypothesis is given in terms of a population parameter:

$H_0$: The data follow a specified distribution.
$H_A$: The data do not follow the specified distribution. (4-5)

For a given confidence level 100*(1-$\alpha$)%, the rejection region is:

$$\chi^2 > \chi^2_{0(\alpha,n-1)} \tag{4-6}$$

Otherwise, it fails to be rejected because the distribution is not significantly different from the theoretical value based on the given confidence level. Here, $\chi^2_{0(\alpha,n-1)}$ is the reference value.

In order to verify whether the distribution of the simulated data set is significantly different as Keepin's theoretical values or not, the total decay time is divided into several periods based on the dominance of different groups. Ten data sets are randomly generated in saturation mode, and their $\chi^2$ values are listed below. The confidence level is set to 95%.

Table 4.1 Chi-square Test between Poisson Data and Keepin's Data ($\alpha$ =0.05)

| Time (s) | n | $\chi^2_{0(\alpha,n-1)}$ | 1# | 2# | 3# | 4# | 5# | 6# | 7# | 8# | 9# | 10# |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0-1 | 40 | 55.76 | 45.39 | 45.35 | 27.06 | 40.11 | 49.29 | 46.27 | 35.03 | 27.49 | 23.67 | 24.00 |
| 1-2 | 40 | 55.76 | 30.17 | 26.29 | 40.58 | 38.82 | 34.10 | 47.30 | 45.06 | 66.27 | 42.58 | 38.43 |
| 2-4 | 80 | 101.88 | 68.25 | 71.74 | 62.42 | 82.81 | 93.95 | 86.86 | 81.90 | 65.80 | 78.73 | 102.42 |
| 4-6 | 80 | 101.88 | 74.68 | 79.63 | 90.95 | 82.03 | 77.17 | 80.42 | 76.77 | 73.49 | 64.19 | 71.64 |
| 6-8 | 80 | 101.88 | 92.87 | 103.24 | 97.02 | 81.36 | 82.57 | 74.53 | 90.50 | 79.60 | 91.15 | 74.94 |
| 8-10 | 80 | 101.88 | 89.38 | 65.67 | 98.36 | 86.19 | 102.51 | 70.30 | 61.24 | 89.85 | 76.36 | 66.81 |
| 10-40 | 60 | 79.08 | 65.12 | 60.74 | 55.32 | 47.62 | 66.94 | 55.57 | 38.66 | 71.85 | 76.87 | 53.58 |
| 40-80 | 80 | 101.88 | 66.19 | 95.37 | 88.81 | 93.19 | 71.44 | 67.93 | 90.42 | 69.35 | 83.66 | 83.84 |
| 80-300 | 80 | 101.88 | 79.89 | 63.17 | 64.27 | 87.03 | 84.36 | 97.40 | 68.34 | 83.82 | 64.80 | 93.20 |

From the $\chi^2$ values in the table, most of them are smaller than $\chi^2_{0(\alpha,n-1)}$. Very small parts of them are larger than the reference values, but their occurrence probabilities are very small. These data set show that values generated by Eq.(4-3) is not significantly different from Keepin's theoretical values with 95% confidence level.

## 4.2 Algorithm I --- Matrix Inverse Method

A Matrix Inverse method for the delayed neutron problem was first proposed by Keepin [9], later developed by Waldo [21]. It is assumed that the delayed neutron count as a function of time could be expressed by the sum of six exponentially decaying groups.

To find the DN parameters for each group, let $z(t_i)$ represent the difference between the measured count and the best fitted count at time $t_i$, let $A_j$ and $\lambda_j$ be the initial guessed yield and decay constant for group j, and let $y(t_i)$ be the measured counts in channel i, thus:

$$z(t_i)=y(t_i) - P\Delta t_i \sum_{j=1}^{n} A_j e^{-\lambda_j t_i} \tag{4-7}$$

We seek values $A_j$' and $\lambda_j$' that will improve the fit of the experimental data, the new values are expressed as follows:

$$\begin{cases} A_j^{'}=A_j+\Delta A_j \\ \lambda_j^{'}=\lambda_j+\Delta\lambda_j \end{cases} \tag{4-8}$$

Substitute Eq. (4-7) and (4-8) into Eq. (4-1), the new objective function can be expressed as:

$$E^2=\sum_{i=1}^{m} w_i[z(t_i) - P\Delta t_i \sum_{j=1}^{n} e^{-\lambda_j t_i}(\Delta A_j - A_j \Delta\lambda_j t_i)]^2 \tag{4-9}$$

where the squares of the differences are weighted by the inverse of their variances. Assuming that the counts are Poisson distributed, then the variance is just the count itself. The weighting factor can be expressed as the inverse of the measured value

$y(t_i)$ so, $w_i = 1/y_i$ and the total number of time channels are m. Usually, 300 seconds counting time is enough to collect the neutrons even from long-lived group precursors.

To optimize the parameters, the above equation can be differentiated with respect to $\Delta A_j$ and $\Delta\lambda_j$ , and the result set to zero (see Waldo's dissertation, 1981):

$$\frac{\partial E^2}{\partial \Delta A_l} = 0 = -\sum_{i=1}^{m} w_i[z(t_i) - (P\Delta t_i)^2 \sum_{j=1}^{n} e^{-\lambda_j t_i}(\Delta A_j - A_j \Delta\lambda_j t_i)][e^{-\lambda_l t_i}] \qquad (4\text{-}10)$$

and

$$\frac{\partial E^2}{\partial \Delta \lambda_l} = 0 = -\sum_{i=1}^{m} w_i[z(t_i) - (P\Delta t_i)^2 \sum_{j=1}^{n} e^{-\lambda_j t_i}(\Delta A_j - A_j \Delta\lambda_j t_i)][e^{-\lambda_l t_i}(-A_l t_i)] \qquad (4\text{-}11)$$

The solution of $\Delta A_l$ and $\Delta\lambda_l$ can be converted into the matrix notation and by inverting and solving the matrix equations to find $\Delta A_l$ and $\Delta\lambda_l$. Formally, let

$$H(l, j) = P\sum_{i=1}^{m} \Delta t_i w_i e^{-(\lambda_l + \lambda_j)t_i} \qquad (4\text{-}12)$$

and

$$D(l) = \sum_{i=1}^{m} w_i z(t_i) e^{-\lambda_l t_i} \qquad (4\text{-}13)$$

The matrix form can be expressed as:

$$\begin{pmatrix} . \\ . \\ D(l) \\ . \\ . \end{pmatrix} = \begin{pmatrix} & . & \\ & . & \\ \ldots & H(l, j) & \ldots \\ & . & \\ & . & \end{pmatrix} \times \begin{pmatrix} . \\ . \\ \Delta A_j \\ . \\ . \end{pmatrix} \qquad (4\text{-}14)$$

Here, the two differentiation equations are dealt with separately and assumed that $\Delta\lambda_j$ is equal to zero in Eq. (4-10). Similarly, $\Delta A_j$ is assumed as zero when solving for $\Delta\lambda_j$ in Eq. (4-11). The matrices **H** and **G** are coefficient matrices.

$$G(l, j) = P\sum_{i=1}^{m} \Delta t_i w_i t_i^2 (-A_j) e^{-(\lambda_l+\lambda_j)t_i} \tag{4-15}$$

and

$$E(l) = \sum_{i=1}^{m} w_i z(t_i) t_i e^{-\lambda_l t_i} \tag{4-16}$$

Similarly,

$$\begin{pmatrix} . \\ . \\ E(l) \\ . \\ . \end{pmatrix} = \begin{pmatrix} & . & \\ & . & \\ \ldots & G(l, j) & \ldots \\ & . & \\ & . & \end{pmatrix} \times \begin{pmatrix} . \\ . \\ \Delta\lambda_j \\ . \\ . \end{pmatrix} \tag{4-17}$$

If the matrices **H** and **G** are nonsingular, it is easy to get the results of $\Delta A_j$ and $\Delta\lambda_j$ by inverse the matrices H(l, j) and G(l, j), as did Waldo [21] before. Using Eq. (4-8) to get the new values of delayed neutron yields and decay constants for different groups by an iteration method until they fit the experimental results very well. They are supposed to converge to the unique ("best fit") values $A_j$ and $\lambda_j$ (j=1, 2…6).

Sometimes, if the coefficients matrix **H** or **G** is ill conditioned or singular, the above equations will have very unstable solutions or no solutions at all. In this case, another method called singular value decomposition may be a good way to find an appropriate solution.

## 4.3 Singular Value Decomposition for Ill-posed Matrix Inverse Problem

Singular value decomposition is a very useful technique to deal with the problems, such as matrix inversion, over-determined and under-determined systems of linear equations. For linear algebraic equation **A*X=B** (**A** is a m×p design matrix, and **B** is observation vector, **X** is parameter vectors), the singular value decomposition of matrix $\mathbf{A}_{m\times p}$ ($m \geq p$) can be represented by a transformation with $\mathbf{A}=\mathbf{UWV}^{T}$ (see the equation below) [51]. $\mathbf{U}_{m\times m}$ and $\mathbf{V}_{p\times p}$ are orthogonal matrices ($\mathbf{U}^{T}\mathbf{U}=\mathbf{I}$ and $\mathbf{V}^{T}\mathbf{V}=\mathbf{I}$), which are called left and right singular vectors separately. The diagonal elements of matrix **W** are non-negative numbers in descending order, and all off-diagonal elements are zeros. The condition number of a matrix is the ratio of the largest $\omega_i$ to the smallest $\omega_j$. The diagonal elements are the singular values, which are the square roots of the eigenvalues, and the rest of the entries of the matrix are zeros.

$$\begin{pmatrix} & \cdot & \\ & \cdot & \\ \cdots & \mathbf{A} & \cdots \\ & \cdot & \\ & \cdot & \end{pmatrix}_{m\times p} = \begin{pmatrix} & \cdot & \\ & \cdot & \\ \cdots & \mathbf{U} & \cdots \\ & \cdot & \\ & \cdot & \end{pmatrix}_{m\times m} \cdot \begin{pmatrix} \omega_1 & & & \\ & \omega_2 & & \\ & & \cdots & \\ & & & \omega_p \\ & & & \cdots \end{pmatrix}_{m\times p} \cdot \begin{pmatrix} & \cdot & \\ & \cdot & \\ \cdots & \mathbf{V}^{T} & \cdots \\ & \cdot & \\ & \cdot & \end{pmatrix}_{p\times p} \tag{4-18}$$

The solution of **A*X=B** can be computed as $\mathbf{X}=\mathbf{V}.\,[\mathrm{diag}(1/\omega_i)].(\mathbf{U}^{T}.\mathbf{B})$ [51]. The solution can be expressed as:

$$\begin{pmatrix} \cdot \\ \cdot \\ \mathbf{X} \\ \cdot \\ \cdot \end{pmatrix} = \begin{pmatrix} & \cdot & \\ & \cdot & \\ \cdots & \mathbf{V} & \cdots \\ & \cdot & \\ & \cdot & \end{pmatrix} . \begin{pmatrix} \cdots \\ \mathrm{diag}(1/\omega_i) \\ \cdots \end{pmatrix} . \begin{pmatrix} & \cdot & \\ & \cdot & \\ \cdots & \mathbf{U}^{T} & \cdots \\ & \cdot & \\ & \cdot & \end{pmatrix} . \begin{pmatrix} \cdot \\ \cdot \\ \mathbf{B} \\ \cdot \\ \cdot \end{pmatrix} \tag{4-19}$$

Set $\mathbf{W}^{*}$=[diag($1/\omega_i$)], which is the Moore-Penrose inverse of **W**, $u_i$ and $v_i$ are the i<sup>th</sup> column of the matrix **U** and **V**. Then the above equation is equivalent to:

$$\mathbf{X} = \mathbf{V}\mathbf{W}^{*}\mathbf{U}^{T}\mathbf{B} = \sum_{i=1}^{p} \frac{\mathbf{u}_i^{T}\mathbf{B}}{\omega_i}\mathbf{v}_i \qquad (4\text{-}20)$$

If the design matrix $\mathbf{A}_{m\times p}$ is full rank (r=min (m, p)), the solution from the least squares method should be unique. The covariance matrix elements of the estimated parameters Cov(X)=$\sigma^2(\mathbf{A}^T\mathbf{A})^{-1}$ can be derived in the following equation:

$$\mathrm{Cov}(x_j, x_k) = \sigma^2 \sum_{i=1}^{p} \left(\frac{V_{ji}V_{ki}}{\omega_i^2}\right) \qquad (4\text{-}21)$$

Here, $\sigma^2$ is the variance of observation error in the regression model. If some of the singular values $\omega_i$ are significantly smaller than the standard deviation of the error, the calculated results will be very large.

If the design matrix **A** is not full rank (r < min (m, p)), part of the singular values are zeros, and the solution will not be unique. In this case, it is useful to set a small threshold [51]. If the singular value is smaller than this threshold, then, it is set to be zero. Suppose there are only r's singular values left, the new solution equation is listed as:

$$\omega_1 \ge \omega_2 \ge \ldots \ge \omega_r = \omega_{r+1} = \ldots = \omega_p = 0.$$

$$\mathbf{X} = \sum_{i=1}^{r} \frac{\mathbf{u}_i^{T}\mathbf{B}}{\omega_i}\mathbf{v}_i \qquad (4\text{-}22)$$

## 4.4 Algorithm II --- Levenberg-Marquardt Method

Theoretically, if the selected model is appropriate and can adequately reflect the real delayed neutron count rate decay characteristics, the following equation should be satisfied:

$$\mathbf{N}_{\exp}(\mathbf{x}, t) \approx \mathbf{N}_{\text{fit}}(\mathbf{x}, t) \qquad (4\text{-}23)$$

$\mathbf{N}_{\exp}(\mathbf{x}, t)$ and $\mathbf{N}_{\text{fit}}(\mathbf{x}, t)$ indicate the experimental counts and the fitted counts from a designed model at time t for a perfect set of "true" parameters. If a set of satisfactory result is achieved for a proper model, it is possible that the fitted value is very close to the real counting data and only has a slight difference. The relationship between them can be expressed by a Taylor-series expansion of the model at the "true" value.

$$\begin{aligned}
&\mathbf{N}_{\exp}(\mathbf{x},t_i)\\
&=\mathbf{N}_{\text{fit}}(\mathbf{x_0},t_i)+\sum_{j=1}^{p}\left[\frac{\partial\mathbf{N}(\mathbf{x_0},t_i)}{\partial x_j}\right]\Delta x_j+\frac{(\Delta x_j)^2}{2!}\sum_{j=1}^{p}\sum_{k=1}^{p}\left[\frac{\partial^2\mathbf{N}(\mathbf{x_0},t_i)}{\partial x_j\partial x_k}\right]+.... \qquad (4\text{-}24)\\
&\approx\mathbf{N}_{\text{fit}}(\mathbf{x_0},t_i)+\sum_{j=1}^{p}\left[\frac{\partial\mathbf{N}(\mathbf{x_0},t_i)}{\partial x_j}\right]\Delta x_j \qquad (i=1,..., m)
\end{aligned}$$

Here, $\Delta\mathbf{x}$ is the difference between "true" value $\mathbf{x}$ and the assumed value at $\mathbf{x}_0$, m is the number of the measurements, and p is the number of the parameters. The above equation can also be described in matrix form as:

$$\Delta\mathbf{N}^0 = \mathbf{N}_{\exp} - \mathbf{N}_{\text{fit}} \approx \mathbf{J}^0\Delta\mathbf{x} \qquad (4\text{-}25)$$

$\Delta\mathbf{N}^0$ is a vector of the measurement departure from "true" value. $\mathbf{J}^0$ is the Jacobian matrix of the objective function $\mathbf{N}(\mathbf{x},t)$. The elements of matrix $\mathbf{J}^0$ are defined as:

$$J_{ij}^{0}=\sum_{j=1}^{p}\left[\frac{\partial \mathbf{N}(\mathbf{x_0},t_i)}{\partial x_j}\right] \tag{4-26}$$

It can be written as:

$$\mathbf{J}^{0}(x_1,...,x_p)=\begin{bmatrix}\frac{\partial N_1}{\partial x_1} & \cdots & \frac{\partial N_1}{\partial x_p}\\ \cdots & \cdots & \cdots\\ \frac{\partial N_m}{\partial x_1} & \cdots & \frac{\partial N_m}{\partial x_p}\end{bmatrix} \tag{4-27}$$

Here, $$\begin{cases}N_1=N_1(x_1,...,x_p)\\ \vdots\\ N_m=N_m(x_1,...,x_p)\end{cases}$$

The weighting factor is usually used to keep equal weighting influence for all the points. It plays a very important role in the least squares method if the count changes with decay time greatly while still keeping slight offset in relative changes. If the two sides of Eq. (4-25) are multiplied by a weighting factor $\sqrt{w_i}$ where $w_i$ is assumed as the inverse of the observation counts at time $t_i$ for Poisson distribution data, then

$$\mathbf{\Delta N}\approx\mathbf{J\Delta x} \tag{4-28}$$

Here, $\Delta$**N** is a vector of weighted residuals with its elements, and **J** is a corresponding vector of weighted Jacobian matrix which rows are weighted from original matrix $J^0$ with $\Delta N_i=\sqrt{w_i}\Delta N_i^0$.

If the matrix **J** is a square matrix, it is very easy to get the solution by inverting matrix **J**. However, the measurement number of an experiment is usually more than the number of unknown parameters (m > p). Therefore, Eq. (4-23) cannot be solved by matrix inversion method directly if m > p. The following method can solve this problem.

In Eq. (4-24), the approximation value only covers the first derivative of the fitting equation. The weighted residual $r_i$ from the model at time $t_i$ is expressed as:

$$\sum_{i=1}^{m} r_i^2 = \sum_{i=1}^{m} w_i \left( \Delta N_i^0 - \sum_{j=1}^{p} J_{ij}^0 \Delta x_j \right)^2 \tag{4-29}$$

To optimize the above equation, the sum square of the unknown parameters deviation from the "true" values is also added for minimization. It is subjected to the constraint: $\sum_{\mathbf{i=1}}^{\mathbf{p}} \Delta \mathbf{x}_i^2 = \sum_{\mathbf{i=1}}^{\mathbf{p}} \Delta \mathbf{x}_{i0}^2$ , where $\triangle x_i^0$ is the deviation of the "best fit" value from the "true" value. A Lagrange multiplier is applied in new function for optimization ([52], page 434):

$$E^2 = \sum_{i=1}^{m} r_i^2 + \lambda \sum_{j=1}^{p} \Delta x_j^2 = \sum_{i=1}^{m} w_i \left( \Delta N_i^0 - \sum_{j=1}^{p} J_{ij}^0 \Delta x_j \right)^2 + \lambda \sum_{j=1}^{p} (\Delta x_j^2 - \Delta x_{j0}^2) \tag{4-30}$$

For minimization, the derivative of the objective function with respect to the unknown parameters deviation Δx is set to zero.

$$\begin{aligned} &\frac{\partial}{\partial \Delta x_k} \left\{ \sum_{i=1}^{m} w_i \left( \Delta N_i^0 - \sum_{j=1}^{p} J_{ij}^0 \Delta x_j \right)^2 + \lambda \sum_{j=1}^{p} \Delta x_j^2 \right\} = 0 \\ &\Rightarrow \sum_{i=1}^{m} \left\{ 2 w_i \left( \Delta N_i^0 - \sum_{j=1}^{p} J_{ij}^0 \Delta x_j \right) (-J_{ik}^0) \right\} + 2\lambda \Delta x_k = 0 \\ &\Rightarrow \sum_{i=1}^{m} \left\{ \sqrt{w_i} \left( \Delta N_i^0 - \sum_{j=1}^{p} J_{ij}^0 \Delta x_j \right) (\sqrt{w_i} J_{ik}^0) \right\} = \lambda \Delta x_k \end{aligned} \tag{4-31}$$

Set $\sqrt{w_i} J_{ik}^0 = J_{ik}$, and $\Delta N_i = \sqrt{w_i} \Delta N_i^0$, the above equation can be rewritten in matrix form as:

$$\mathbf{\Delta x = (J^T J + \lambda I)^{-1} J^T \Delta N} \tag{4-32}$$

This is Levenberg-Marquardt method [52] [53]. Sometimes, the condition number, which is the matrix norm of the square matrix $\mathbf{J^T J}$, may be very large. In this situation, the matrix $\mathbf{J^T J}$ is ill posed. If this is true, a small random relative perturbation of the value in the matrix will result in a large variation in a linear system. Thus, the regularization constant λ in Levenberg-Marquardt can play a very important role to adjust the condition number. Due to the introduction of diagonal matrix $\lambda\mathbf{I}$ with proper value λ, the matrix item $\mathbf{J^T J} + \lambda\mathbf{I}$ will no longer be singular. In practice, the diagonal elements of $\mathbf{J^T J}$ are sometimes of significantly different magnitudes, and Marquardt [52] suggested

$$\mathbf{\Delta x = [J^T J + \lambda diag(J^T J)]^{-1} J^T \Delta N} \tag{4-33}$$

The Levenberg-Marquardt technique uses a trust region method to shrink the step size to ensure the reduction of sum-of-residuals at each iteration. At each iteration, λ is decreased after each successful step (reduction in objective function) and is increased only when a tentative step increases the objective function. If $E_{k+1}^2 > E_k^2$, increase λ and recalculate $E_{k+1}^2$; otherwise, decrease λ and recalculate $E_{k+1}^2$.

The Levenberg-Marquardt technique combines the Gauss-Newton method and the steepest-descent method. It uses the method of linear descent in early iterations and then gradually switches to the Gauss-Newton approach. It almost always converges to the correct parameter estimates even when the Hessian matrix (defined in Section 4.5) is not positive. It can deal with a near-singularity matrix in a reasonable way to generate accurate predictions even if the optimal weight values are very bad. It restricts the step to stay at the surface of or inside an n-dimensional elliptical (or spherical) trust region. If λ gets small or if the sum of weighted squared residuals gets smaller at each iteration, the algorithm will use the Gauss-Newton algorithm. If λ gets large or if the sum of weighted squared residuals increases at each iteration, the algorithm will use the steepest descent technique. The performance of this algorithm is usually poor in the large-residual case due to slow or no convergence [53]. Asymptotic convergence is only linear-slower than the super-linear convergence rate attained by algorithms for general unconstrained problems, such as Newton or quasi-Newton.

## 4.5 Algorithm III --- Quasi-Newton Method

As mentioned before, the objective function can be expanded into Taylor series:

$$\mathbf{N}(\mathbf{x})=\mathbf{N}(\mathbf{x}_k)+\mathbf{N}'(\mathbf{x}_k)(\mathbf{x}\text{-}\mathbf{x}_k)+\frac{1}{2}(\mathbf{x}\text{-}\mathbf{x}_k)\mathbf{N}''(\mathbf{x}_k)(\mathbf{x}\text{-}\mathbf{x}_k) \tag{4-34}$$

The solution can be found by setting the derivatives to zero.

$$\begin{aligned}&\mathbf{N}'(\mathbf{x})=\mathbf{N}'(\mathbf{x}_k)+\mathbf{N}''(\mathbf{x}_k)(\mathbf{x}\text{-}\mathbf{x}_k)=0\\&\mathbf{x}=\mathbf{x}_k\text{-}[\mathbf{N}''(\mathbf{x}_k)]^{-1}\mathbf{N}'(\mathbf{x}_k)\end{aligned} \tag{4-35}$$

If the function $\mathbf{N}(\mathbf{x})$ is quadratic, the solution can be found in a single step. If it is not quadratic, the solution need to be found be iteration method. $\mathbf{N}''(\mathbf{x}_k)$ is the Hessian matrix. However, Hessian matrix is very difficult to find computationally. Quasi-Newton attempts to produce an estimation of the inverse of Hessian matrix for the next iteration point by Broyden-Fletcher-Goldfarb-Shanno (BFGS) formula [54]. If the point $\mathbf{x}_k$ is far from the optimum, the quadratic function may be poorly approximated. A step length α is introduced to deal with this. A line search is used to find the next point

$$\mathbf{x}_{k+1}=\mathbf{x}_k-\lambda\mathbf{A}_k^{-1}\mathbf{g}_k,\quad \lambda>0$$
$$\text{BFGS:}\quad \mathbf{A}\leftarrow\mathbf{A}-\frac{\mathbf{A}*\mathbf{s}*\mathbf{s}^T*\mathbf{A}}{\mathbf{s}^T*\mathbf{A}*\mathbf{s}}+\frac{\mathbf{y}*\mathbf{y}^T}{\mathbf{y}^T*\mathbf{s}}\qquad\begin{pmatrix}\mathbf{s}=\mathbf{x}_{k+1}-\mathbf{x}_k\\ \mathbf{y}=\mathbf{g}_{k+1}-\mathbf{g}_k\end{pmatrix}\tag{4-36}$$

so that $$\mathbf{N}_{k+1}\le\mathbf{N}_k-\alpha\mathbf{g}_k^T\mathbf{A}_k^{-1}\mathbf{g}_k,\quad \alpha\in(0,0.5)\tag{4-37}$$

The optimization values are achieved when $||\mathbf{g}(\mathbf{x})||=\varepsilon$ (gradient tolerance). Here, **g** is the gradient; $\mathbf{x}_k$ is the parameter set at the k$^{th}$ iteration; λ is the positive step length; **A** is a positive definite approximation of the Hessian matrix obtained by BFGS formula [54]; and $\alpha$ is regularization constant. This routine uses Quasi-Newton [54] method with a finite difference gradient to help locate a minimum. If λ=1 and **A** is the exact Hessian matrix, the above algorithm becomes Newton method. If **A** is the identity matrix, it becomes the steepest descent method. The quasi-Newton method doesn't require calculation of the inverse of the second derivatives of the weight estimates. The quasi-Newton method develops an approximation to the Hessian during training and therefore adapts even more effectively to changing condition numbers. Both of the above methods (Levenberg-Marquardt and quasi-Newton) can find local optima, and they are not guaranteed to find a global optimum. Quasi-Newton converges at a faster asymptotic rate than Levenberg-Marquardt in the large-residual case, but slower than Newton-based methods. Its behavior on early iterations (before the iteration reach a neighborhood of the solution) may be inferior to Levenberg-Marquardt. At the beginning, quasi-Newton takes steps in the gradient descent direction and uses an identity Hessian matrix instead. When the quasi-Newton method reaches a local minimum point, the error estimate will stay a local minimum. If the optimization technique approaches a saddle point, the error estimate will keep a saddle point for an extended period of time. Same as Levenberg-Marquardt method, this algorithm results in a deviance objective function. One of the drawbacks of this method is that the quasi-Newton method might generate inadequate approximation of Hessian matrix. The routines of these two methods can be found from IMSL math library [55].

Once the necessary minimum data have been searched, the estimated covariance matrix **C** of the standard errors is the inverse of Hessian matrix **h** at the final fitted parameters. It is not the inverse of the estimated Hessian matrix **A**.

$$\mathbf{C}=\mathbf{h}^{-1} \tag{4-38}$$

Hessian matrix is described as below:

$$\mathbf{h}(\mathrm{x}_1,\ldots,\mathrm{x}_\mathrm{p})=\begin{bmatrix} \dfrac{\partial^2\mathbf{N}}{\partial\mathrm{x}_1\partial\mathrm{x}_1} & \cdots & \dfrac{\partial^2\mathbf{N}}{\partial\mathrm{x}_1\partial\mathrm{x}_\mathrm{p}} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial^2\mathbf{N}}{\partial\mathrm{x}_\mathrm{p}\partial\mathrm{x}_1} & \cdots & \dfrac{\partial^2\mathbf{N}}{\partial\mathrm{x}_\mathrm{p}\partial\mathrm{x}_\mathrm{p}} \end{bmatrix} \tag{4-39}$$

Here, **N**=**N** ($\mathrm{x}_1$, $\mathrm{x}_2$, …, $\mathrm{x}_\mathrm{p}$) is a function of the delayed neutron fitting parameters.

## 4.6 Results and Analysis

### 4.6.1 Fitted Results from Saturation Irradiation

As described earlier in Section 4.1, simulated data was generated and the three algorithms were used to extract parameters from this data. The only variation for this data is from the Poisson distribution. No other measurement errors are considered. Here, twelve different random seeds are randomly selected to generate twelve sets of simulated “experimental data”. In different numerical algorithms, the simulated “experimental data sets” generated from same random numbers are also the same for comparison. The theoretical data are generated based on the values provided by Keepin with the same initial count rate (4E+5 cps as an example) in all these three algorithms. If the initial count rate is too low, the relative count variation from Poisson data will be very high. If it is too large, the dead time problem in the real experiment will be very severe. This count rate is a reasonable value. The computation programs written in FORTRAN codes are listed in Appendix A.1, A.2, and A.3.

The calculated results in a saturation mode using different numerical algorithms mentioned in the previous sections are shown in tables below (Table 4.2, 4.3, and 4.4). $A_i/A_i$’ and $R_i/R_i$’ are the ratios of calculated relative yields and decay constants to Keepin’s recommended values [10] for group i (i = 1, 2 … 6) respectively. The symbol $E^2$ is the sum of the weighted squared differences between the simulated “experimental data” and the theoretical data. If the original parameters were returned exactly by the algorithm, the fitted ratio will be equal to 1.0. The purpose of this comparison is to determine how well these algorithms can return the input parameters.

Table 4.2 Fitted Results from Matrix Inversion with SVD Algorithm (Saturation Mode)

| seed | -6295779 | -389279151 | 738164549 | -1778841095 | 1234 | -1031501663 | -1141185387 | 774385225 | 397015485 | 534354673 | 4250 | -25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $A_1/A_1'$ | 0.99134 | 1.00606 | 1.00571 | 1.00634 | 0.99556 | 0.99260 | 1.00596 | 1.01054 | 0.99947 | 0.99363 | 0.98788 | 0.99942 |
| $A_2/A_2'$ | 1.00493 | 0.99845 | 0.99669 | 0.99681 | 1.00264 | 1.00440 | 0.99658 | 0.99389 | 0.99959 | 1.00421 | 1.00772 | 1.00135 |
| $A_3/A_3'$ | 0.98753 | 0.99726 | 1.00284 | 1.01028 | 0.99393 | 0.98630 | 1.01244 | 1.02060 | 1.00781 | 0.99058 | 0.97426 | 0.98776 |
| $A_4/A_4'$ | 1.00856 | 1.00824 | 1.00067 | 0.99280 | 1.00064 | 1.01186 | 0.98998 | 0.98236 | 0.99136 | 1.00112 | 1.02022 | 1.01237 |
| $A_5/A_5'$ | 0.93188 | 0.85975 | 0.96871 | 1.01026 | 1.03124 | 0.98209 | 1.03131 | 1.07870 | 1.05136 | 0.99089 | 0.90738 | 0.91772 |
| $A_6/A_6'$ | 1.14866 | 1.47314 | 1.22348 | 1.06739 | 0.75566 | 0.90959 | 0.97250 | 0.77673 | 0.95117 | 0.85037 | 1.09334 | 1.29992 |
| $\lambda_1/\lambda_1'$ | 0.99662 | 0.99966 | 1.00037 | 1.00205 | 0.99982 | 0.99764 | 1.00339 | 1.00346 | 1.00287 | 0.99506 | 0.99626 | 0.99807 |
| $\lambda_2/\lambda_2'$ | 1.00092 | 1.00047 | 1.00055 | 0.99947 | 0.99975 | 1.00042 | 0.99895 | 0.99966 | 0.99915 | 1.00170 | 0.99978 | 1.00162 |
| $\lambda_3/\lambda_3'$ | 1.00398 | 0.98883 | 0.99638 | 0.99529 | 0.99863 | 1.00610 | 0.99939 | 0.98730 | 1.00142 | 0.99489 | 1.01357 | 0.98623 |
| $\lambda_4/\lambda_4'$ | 0.99433 | 1.00689 | 1.00460 | 1.00352 | 1.00334 | 0.99397 | 0.99965 | 1.00937 | 0.99884 | 1.00236 | 0.99006 | 1.01193 |
| $\lambda_5/\lambda_5'$ | 1.09542 | 0.99482 | 0.97010 | 0.95498 | 0.97618 | 1.03508 | 1.00560 | 0.96060 | 0.97750 | 1.09093 | 1.07452 | 0.95182 |
| $\lambda_6/\lambda_6'$ | 0.69251 | 1.05737 | 1.05045 | 1.30784 | 1.01209 | 0.75921 | 0.99180 | 1.17428 | 1.10825 | 0.69556 | 0.84690 | 1.12338 |
| $E^2$ | 645.14 | 667.66 | 565.97 | 653.44 | 668.86 | 616.75 | 568.11 | 588.96 | 626.69 | 664.55 | 598.36 | 600.94 |

Table 4.3 Fitted Results from Levenberg-Marquardt Algorithm (Saturation Mode)

| seed | -6295779 | -389279151 | 738164549 | -1778841095 | 1234 | -1031501663 | -1141185387 | 774385225 | 397015485 | 534354673 | 4250 | -25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $A_1/A_1'$ | 0.96866 | 1.00789 | 1.02352 | 1.00105 | 0.99465 | 1.06648 | 0.99186 | 0.97128 | 1.02258 | 1.04708 | 0.99522 | 0.98823 |
| $A_2/A_2'$ | 0.99848 | 0.99989 | 0.99356 | 0.99922 | 1.00408 | 0.99157 | 0.99836 | 1.01071 | 0.98327 | 1.00266 | 1.00656 | 0.99610 |
| $A_3/A_3'$ | 0.96638 | 1.01132 | 1.01693 | 0.99825 | 0.99457 | 0.95238 | 1.01620 | 0.91100 | 0.84078 | 1.06866 | 1.06216 | 1.03629 |
| $A_4/A_4'$ | 1.03065 | 0.99974 | 1.00967 | 0.99753 | 0.99840 | 1.01439 | 0.98770 | 0.85553 | 1.01348 | 0.97599 | 0.99202 | 1.01884 |
| $A_5/A_5'$ | 0.84899 | 0.82692 | 0.95981 | 1.09398 | 0.95412 | 0.64688 | 0.97407 | 1.29219 | 1.26829 | 0.82334 | 0.45779 | 0.92504 |
| $A_6/A_6'$ | 1.37227 | 1.53010 | 1.09719 | 0.90741 | 0.94495 | 1.94759 | 1.15188 | 1.97190 | 0.84280 | 1.00958 | 2.08246 | 1.05273 |
| $\lambda_1/\lambda_1'$ | 0.99070 | 1.00192 | 1.00727 | 1.00036 | 0.99875 | 1.02121 | 0.99875 | 0.98844 | 1.01274 | 1.01314 | 0.99904 | 0.99728 |
| $\lambda_2/\lambda_2'$ | 0.99191 | 1.00127 | 1.00244 | 0.99968 | 1.00097 | 1.00905 | 0.99676 | 0.99570 | 1.00334 | 1.01366 | 1.00294 | 0.99549 |
| $\lambda_3/\lambda_3'$ | 0.96928 | 1.00550 | 0.99778 | 0.99354 | 1.00317 | 0.98216 | 0.99877 | 0.97012 | 0.93191 | 1.05133 | 1.04646 | 0.99342 |
| $\lambda_4/\lambda_4'$ | 0.98507 | 1.01043 | 1.02176 | 0.99714 | 1.00506 | 0.98109 | 0.99995 | 0.87466 | 0.91829 | 1.05134 | 1.04134 | 1.04289 |
| $\lambda_5/\lambda_5'$ | 1.09584 | 0.98863 | 1.04088 | 1.00128 | 0.94131 | 0.82781 | 0.96545 | 0.53909 | 0.91294 | 1.32339 | 0.97555 | 1.08175 |
| $\lambda_6/\lambda_6'$ | 0.62948 | 1.00469 | 1.10698 | 1.71523 | 0.87051 | 0.61389 | 0.91643 | 0.69969 | 1.56007 | 0.49297 | 0.65009 | 1.27612 |
| $E^2$ | 643.18 | 662.67 | 564.22 | 652.18 | 667.67 | 605.84 | 567.81 | 574.31 | 621.39 | 659.18 | 590.86 | 595.10 |

Table 4.4 Fitted Results from Quasi-Newton Algorithm (Saturation Mode)

| seed | -6295779 | -389279151 | 738164549 | -1778841095 | 1234 | -1031501663 | -1141185387 | 774385225 | 397015485 | 534354673 | 4250 | -25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $A_1/A_1'$ | 1.15059 | 1.03515 | 1.01249 | 1.00110 | 0.97480 | 0.94484 | 0.97906 | 0.90862 | 1.01042 | 1.09663 | 1.00127 | No solution is found. |
| $A_2/A_2'$ | 1.24255 | 0.98662 | 0.98329 | 1.01212 | 1.00478 | 0.87845 | 0.98534 | 0.94542 | 0.97158 | 1.03342 | 1.01118 | |
| $A_3/A_3'$ | 0.71069 | 1.11547 | 1.00518 | 0.95058 | 1.00129 | 0.73558 | 1.00366 | 0.85336 | 0.83078 | 1.14558 | 1.07053 | |
| $A_4/A_4'$ | 0.77189 | 0.94142 | 0.99956 | 1.03184 | 0.97662 | 0.65531 | 0.97553 | 0.80121 | 1.00142 | 0.98166 | 0.99079 | |
| $A_5/A_5'$ | 3.22380 | 0.23949 | 0.95163 | 1.19910 | 0.85107 | 1.32360 | 0.97514 | 1.20443 | 1.25321 | 0.88731 | 0.31159 | |
| $A_6/A_6'$ | 4.13037 | 2.62317 | 1.08886 | 0.96053 | 1.35363 | 2.58144 | 1.08914 | 1.84035 | 0.83278 | 0.83292 | 2.52245 | |
| $\lambda_1/\lambda_1'$ | 0.96788 | 1.01663 | 1.00709 | 0.99492 | 0.99300 | 1.01985 | 0.99880 | 0.98848 | 1.01274 | 1.01909 | 0.99958 | |
| $\lambda_2/\lambda_2'$ | 0.96365 | 1.01627 | 1.00225 | 0.99305 | 0.99940 | 1.00591 | 0.99681 | 0.99575 | 1.00334 | 1.01996 | 1.00349 | |
| $\lambda_3/\lambda_3'$ | 0.72943 | 1.08819 | 0.99680 | 0.95007 | 1.01162 | 0.92788 | 0.99911 | 0.97072 | 0.93191 | 1.08407 | 1.04939 | |
| $\lambda_4/\lambda_4'$ | 0.60875 | 1.09421 | 1.02080 | 0.95764 | 1.00240 | 0.82228 | 1.00085 | 0.87585 | 0.91829 | 1.07806 | 1.04205 | |
| $\lambda_5/\lambda_5'$ | 0.32127 | 0.75268 | 1.03869 | 0.94565 | 0.82757 | 0.37204 | 0.97723 | 0.54074 | 0.91294 | 1.34928 | 0.82173 | |
| $\lambda_6/\lambda_6'$ | 0.54372 | 0.72374 | 1.10901 | 1.82089 | 0.78057 | 0.53719 | 0.93763 | 0.70058 | 1.56007 | 0.51102 | 0.61850 | |
| $E^2$ | 640.52 | 660.10 | 564.22 | 651.74 | 666.87 | 605.57 | 567.81 | 574.31 | 621.39 | 658.86 | 590.84 | |

The sum of weighted squared difference $E^2$ is a useful indication of how closely the data is fit by these algorithms, and a fitted ratio near 1.0 is an indicator of solution stability. From the results in Table 4.2, 4.3 and 4.4, it is very clear that the above numerical algorithms can produce fitted results with similar $E^2$ values (the sum of weighted squared differences) in most cases. However, the quasi-Newton method cannot find a solution for some data sets. Among those solvable data sets, the solutions are not stable, especially for short half-lived groups (Group-5 and -6) because the ratios are not near 1.0. For the same data set with similar $E^2$ value, some extracted parameters are significantly different, meaning that the solution is not stable. What the algorithms found are not the real global optima. For same simulated input data, the calculated $E^2$ values using Levenberg-Marquardt (LM) and quasi-Newton (QN) method can often achieve a relatively smaller sum of weighted squared difference ($E^2$) compared to the Matrix Inverse method combined with singular value decomposition (SVD). However, the extracted parameters from LM and QN have much larger departure from the original input parameters. For all of the methods above, the biggest difference always happens to Group-5 and -6 parameters (the shorter half-lived groups). Failure of the quasi-Newton algorithm to converge means that the estimated Hessian matrix using this method is essentially singular and cannot be inverted for the solution.

The fitted curves using the results from three different numerical algorithms are plotted in the Figure 4.3 below. The Poisson distributed data set (seed=774385225) is randomly selected and is plotted for comparison. The initial count rates from these data sets are Poisson distributed around the same value (4E+5 cps) as an example. From the plot, all of the fitted results fit the simulated experimental data (Poisson data) very well and it is hard to distinguish any significant difference between them.

The distribution of extracted results from many data sets can provide more detailed information. A solution with a small deviation is considered to be a stable and convincing solution. Figure 4.4 shows the standard deviation of fitted group parameter ratios. Since the Levenberg-Marquardt method always converges with small $E^2$ values, this algorithm is chosen to evaluate the solution distribution. The data sources are from twelve data sets mentioned in Table 4.3. The standard deviation shows how stable these data are clustered around the ideal value 1.0.

From the plots, it is obvious that for shorter half-lived groups, Group-5 and -6, the calculated results have larger uncertainty in saturation mode. The variation of the yields is larger than those of the decay constants for same group. Longer half-lived groups, Group-1, -2, -3, and 4, have smaller variations and the fitted results are relatively more stable than the short half-lived groups (Group-5 and -6). This is because the total counts from shorter-lived groups are relatively much smaller than longer-lived groups, and this increase the uncertainties.

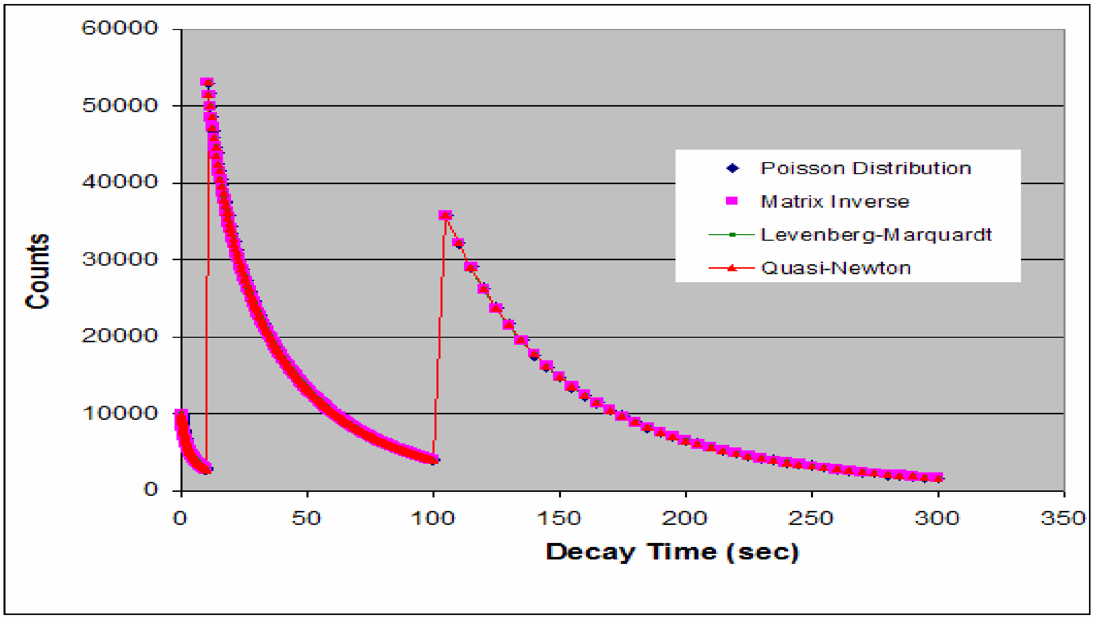


Figure 4.3 Simulation Results from Three Algorithms

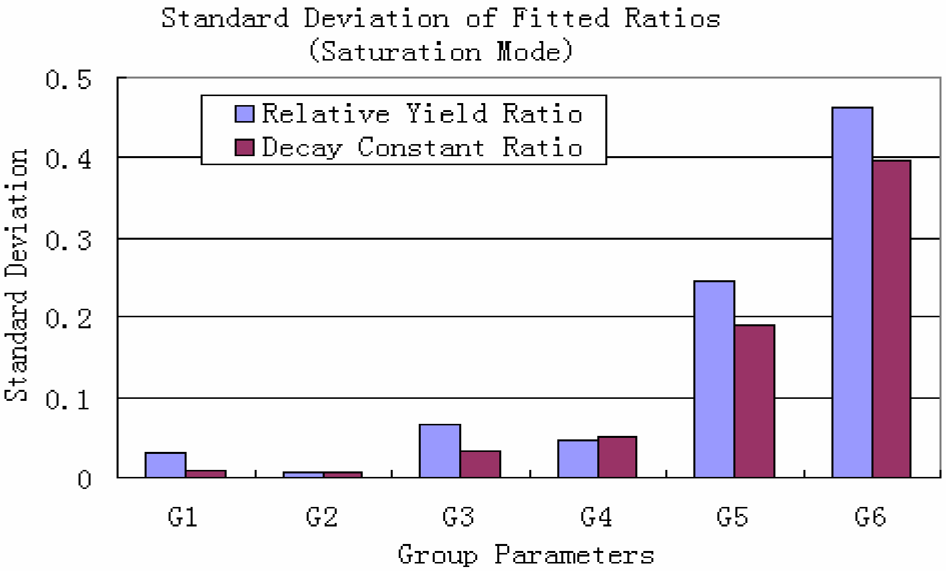


Figure 4.4 Standard Deviations of Fitted Group Parameter Ratios from Levenberg-Marquardt Algorithm (Saturation Mode)

### 4.6.2 Fitted Results from Pulse Irradiation

For single pulse irradiation, the objective function is described as:

$$E^2 = \sum_{i=1}^{m} \frac{1}{y_i}\left( y_i - P\Delta t_i \sum_{j=1}^{n} \lambda_j A_j e^{-\lambda_j t_i} \right)^2 \tag{4-40}$$

Here, $y_i$ is the $i^{th}$ experimental count, $\Delta t_i$ is the dwell time for the $i^{th}$ measurement. The fitted results from pulse irradiation are also calculated and listed in Table 4.5 (Levenberg-Marquardt) and Table 4.6 (quasi-Newton). Ten data sets were used for calculation. The standard deviation of different group parameters from ten different data sets analyzed using Levenberg-Marquardt method is shown in Figure 4.5. The simulated data sets are generated from Keepin's parameters as described earlier and both saturated and pulse data are set so that the initial count rate is the same (4E+5 cps). Then, the data are randomized so as to be Poisson distributed including this first channel.

In pulse mode, from Figure 4.5 it is clear that the fitted data for short-lived groups (Group-5 and -6) are relatively more stable compared to that in saturation mode (Figure 4.4), but for long-lived group (Group-1), the fitted data in pulse mode are very unstable.

Table 4.5 Fitted Results from Levenberg-Marquardt Algorithm (Pulse Mode)

| seed | -6295779 | -389279151 | 738164549 | -1778841095 | 1234 | -1031501663 | -1141185387 | 774385225 | 397015485 | 534354673 |
|---|---|---|---|---|---|---|---|---|---|---|
| $A_1/A_1'$ | 0.81490 | 1.36500 | 1.04563 | 0.87687 | 0.97636 | 1.23462 | 1.03147 | 0.98429 | 1.14010 | 1.37309 |
| $A_2/A_2'$ | 1.00132 | 0.99770 | 0.97929 | 0.99075 | 1.01946 | 0.94936 | 1.00334 | 1.01132 | 0.93937 | 1.00323 |
| $A_3/A_3'$ | 0.96291 | 1.06923 | 1.01995 | 0.98815 | 0.99457 | 0.96104 | 1.04598 | 0.99134 | 0.89891 | 0.97406 |
| $A_4/A_4'$ | 1.02054 | 0.95150 | 0.99756 | 1.01986 | 0.98680 | 1.01884 | 0.98322 | 0.98881 | 1.01394 | 0.99111 |
| $A_5/A_5'$ | 0.94495 | 0.90861 | 1.00404 | 1.07026 | 0.99343 | 0.95208 | 0.99728 | 1.03128 | 1.04922 | 0.93862 |
| $A_6/A_6'$ | 1.22837 | 1.26469 | 1.00279 | 0.78314 | 1.07820 | 1.26892 | 0.93597 | 1.01553 | 0.82656 | 1.71797 |
| $\lambda_1/\lambda_1'$ | 0.92846 | 1.11682 | 1.02508 | 0.95781 | 0.98544 | 1.11183 | 1.02238 | 0.99183 | 1.09581 | 1.10359 |
| $\lambda_2/\lambda_2'$ | 0.97300 | 1.05397 | 0.99675 | 0.97598 | 1.00901 | 1.00769 | 1.00591 | 1.00731 | 1.01335 | 1.04499 |
| $\lambda_3/\lambda_3'$ | 0.94920 | 1.08739 | 0.98918 | 0.95330 | 1.01292 | 0.95445 | 1.03158 | 1.00857 | 0.93944 | 1.00804 |
| $\lambda_4/\lambda_4'$ | 0.97826 | 1.02876 | 1.00572 | 0.99016 | 1.00208 | 0.98001 | 1.01729 | 0.98888 | 0.97333 | 0.96080 |
| $\lambda_5/\lambda_5'$ | 0.94750 | 0.92654 | 0.99712 | 1.04609 | 0.95728 | 0.91595 | 1.03478 | 0.97059 | 1.02691 | 0.77468 |
| $\lambda_6/\lambda_6'$ | 0.90676 | 0.93771 | 1.01647 | 1.12619 | 0.95778 | 0.89195 | 1.02346 | 0.98838 | 1.08102 | 0.80229 |
| $E^2$ | 643.73 | 660.28 | 567.11 | 653.34 | 667.11 | 604.94 | 569.56 | 577.77 | 621.02 | 656.68 |

Table 4.6 Fitted Results from Quasi-Newton Algorithm (Pulse Mode)

| seed | -6295779 | -389279151 | 738164549 | -1778841095 | 1234 | -1031501663 | -1141185387 | 774385225 | 397015485 | 534354673 |
|---|---|---|---|---|---|---|---|---|---|---|
| $A_1/A_1'$ | 0.59946 | 1.29956 | 1.18268 | 0.89182 | 0.96908 | 1.28058 | 1.02294 | 0.93476 | 1.15876 | 1.25563 |
| $A_2/A_2'$ | 0.97159 | 0.95006 | 0.95361 | 1.03402 | 1.01199 | 0.89239 | 1.00272 | 0.99498 | 0.95474 | 0.91770 |
| $A_3/A_3'$ | 0.87905 | 1.01784 | 1.02775 | 1.03031 | 0.99022 | 0.88543 | 1.04432 | 0.93422 | 0.91361 | 0.89089 |
| $A_4/A_4'$ | 1.01316 | 0.90624 | 0.96744 | 1.06570 | 0.97067 | 0.97176 | 0.98351 | 0.97351 | 1.03052 | 0.90667 |
| $A_5/A_5'$ | 0.94289 | 0.86541 | 0.97676 | 1.11796 | 0.97665 | 0.91208 | 0.99687 | 1.03460 | 1.06638 | 0.85871 |
| $A_6/A_6'$ | 1.56164 | 1.20517 | 0.92944 | 0.81686 | 1.10370 | 1.35231 | 0.94977 | 1.13483 | 0.84008 | 1.57170 |
| $\lambda_1/\lambda_1'$ | 0.81393 | 1.11676 | 1.07937 | 0.94626 | 0.98530 | 1.14262 | 1.01924 | 0.97725 | 1.09581 | 1.10347 |
| $\lambda_2/\lambda_2'$ | 0.93308 | 1.05388 | 1.01937 | 0.97235 | 1.01072 | 1.01868 | 1.00416 | 1.00243 | 1.01336 | 1.04491 |
| $\lambda_3/\lambda_3'$ | 0.86348 | 1.08703 | 1.02158 | 0.94907 | 1.02101 | 0.94029 | 1.02792 | 0.98687 | 0.93944 | 1.00787 |
| $\lambda_4/\lambda_4'$ | 0.92445 | 1.02853 | 1.02388 | 0.98896 | 1.00336 | 0.96145 | 1.01571 | 0.96219 | 0.97333 | 0.96072 |
| $\lambda_5/\lambda_5'$ | 0.79036 | 0.92607 | 1.02894 | 1.04590 | 0.94533 | 0.85331 | 1.02895 | 0.90079 | 1.02691 | 0.77456 |
| $\lambda_6/\lambda_6'$ | 0.82703 | 0.93749 | 1.03736 | 1.13255 | 0.94764 | 0.85982 | 1.01871 | 0.94535 | 1.08102 | 0.80226 |
| $E^2$ | 642.07 | 660.28 | 566.63 | 653.21 | 666.97 | 604.27 | 569.51 | 577.13 | 621.02 | 656.68 |

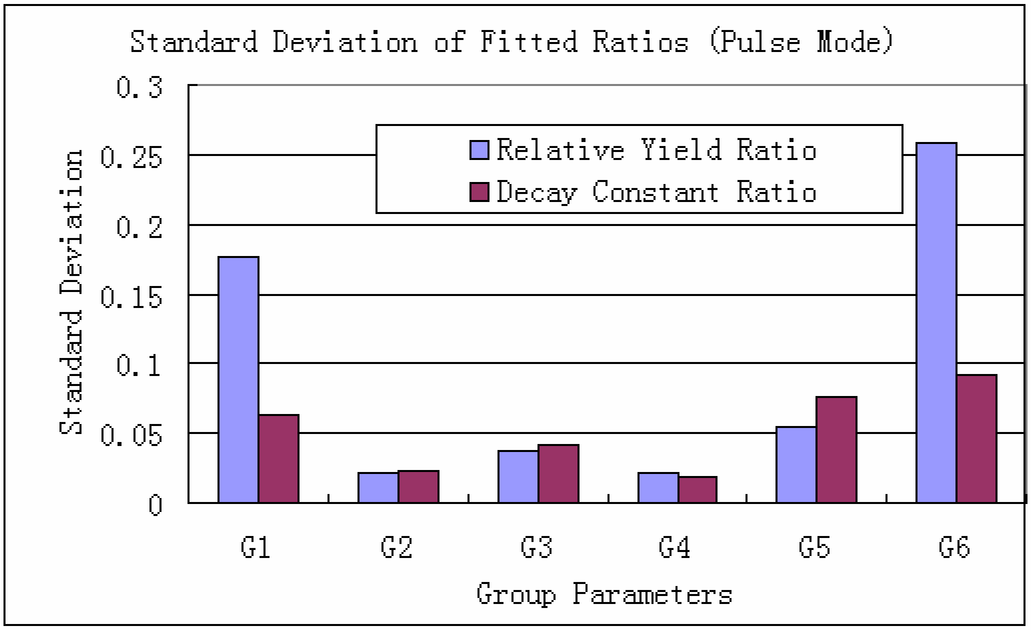


Figure 4.5 Standard Deviations of Fitted Group Parameter Ratios from Levenberg-Marquardt Algorithm (Pulse Mode)

## 4.7 Problem Analysis

In order to make clear why the solution is so variable even for the same Poisson distributed data, the condition number, which is a useful indicator for unstable or ill-conditioned matrix, is analyzed. The condition number of matrix **A**, is the product of $||\mathbf{A}||*||\mathbf{A}^{-1}||$. If we use the usual Euclidean norm on vectors and the associated matrix norm, then the condition number is the ratio of the largest singular value of matrix to the smallest singular value of matrix. The ill-conditioned problem does not have an existing, unique, or stable solution. Any "noise" such as round off errors can be significantly amplified and may cause a physically meaningless solution. If the design matrix is singular, or close to singular with large condition numbers, it is difficult to get the inverse matrix directly through traditional Gaussian elimination or LU decomposition methods. Even if the final results can be obtained, the parameters usually have very large magnitudes and are unstable when the fitted function is evaluated. Therefore, direct derivation is almost impossible when the normal equations encounter a zero pivot. If condition number is very large, the solutions are unstable with respect to small changes of input data. In an ill-posed problem, the singular values in Eq. (4-17) gradually converge toward zero and become so small that the solution from ordinary least squares becomes unstable.

To verify this, a small perturbation of ΔB is inserted in the original problem **A*****X**=**B**. The solution of the perturbed system is: **X***=**X**+Δ**X**, and the condition number of matrix **A** is

$$\begin{aligned}\text{cond}(\mathbf{A}) = \|\mathbf{A}\|*\|\mathbf{A}^{-1}\| &= \left(\frac{\|\mathbf{\Delta X}\| / \|\mathbf{X}\|}{\|\mathbf{\Delta B}\| / \|\mathbf{B}\|}\right) \\ &= \max\left(\frac{\text{Relative error of solution}}{\text{Relative error of inputs}}\right)\end{aligned} \qquad (4\text{-}41)$$

Thus, any relative changes in the input value **B** will be multiplied by the condition number and generate the relative changes in the output solutions **X**. If machine precision is ε (epsilon), the order of relative error will be ε*cond(**A**), which is usually

2.2204E-016*cond(**A**)

A condition number is said to be too large if it is larger or comparable to

1/ε = 4.5036E+015

In this condition, no significant digit in the result is reliable!

Taking saturation mode as an example, the condition number of the corresponding matrices used in different the algorithms are listed in Table 4.7. Ten data sets mentioned in previous simulation are used to compare. In the table below, **H** and **G** are the matrices used in the Matrix Inverse method; **J** is Jacobian matrix; **h** is the Hessian matrix. The condition numbers listed below are the values in the first iterative loop before regularization.

Table 4.7 Condition Numbers of the Matrices Used in Different Algorithms (Saturation Mode)

| | Cond(**H**) | Cond(**G**) | Cond($\mathbf{J}^{T*}\mathbf{J}$) | Cond(**h**) |
|---|---|---|---|---|
| Data 1# | 2.39E+04 | 7.13E+09 | 1.83E+12 | 2.72E+13 |
| Data 2# | 5.65E+05 | 2.08E+07 | 9.47E+11 | 1.53E+12 |
| Data 3# | 6.10E+03 | 5.27E+09 | 6.46E+10 | 9.43E+10 |
| Data 4# | 4.52E+03 | 7.10E+07 | 1.27E+13 | 1.07E+10 |
| Data 5# | 5.89E+03 | 8.89E+07 | 2.05E+11 | 3.76E+10 |
| Data 6# | 1.31E+04 | 6.04E+10 | 2.48E+13 | 2.72E+11 |
| Data 7# | 6.70E+03 | 5.35E+07 | 1.37E+11 | 1.01E+10 |
| Data 8# | 1.23E+04 | 4.62E+13 | 8.78E+17 | 4.76E+13 |
| Data 9# | 5.55E+03 | 6.77E+07 | 1.86E+11 | 6.93E+09 |
| Data 10# | 1.82E+04 | 6.06E+10 | 1.21E+12 | 1.45E+09 |

Most of the condition numbers are very high in the above matrices before regularization. In the Matrix Inverse method without singular value decomposition, condition number of matrix G is as high as 4.62E13 for data set 8#. Only two significant digits are reliable in this situation. Obviously, the matrix is severely ill conditioned. After several iterations in the algorithm, some condition numbers even become infinity. With the help of regularization, the condition numbers can be reduced. In LM algorithm, the item $\mathbf{J}^T\mathbf{J}$ is replaced with $\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I}$ to avoid the ill-posed problem as long as $\lambda$ is not too small. In the quasi-Newton algorithm, the estimated Hessian matrix is multiplied by a regularization parameter to adjust the condition number. Due to the diversity of the regularization techniques, the final results can be different even for the same data set. It is very important to choose an appropriate regularization number to reduce the condition number as well as to keep the bias as small as possible. If the value is too small, the solution will still not be able to converge. If it is too large, the solution of the parameters will be predominated with bias. A new numerical method introduced in the next section will improve this situation and produce a better solution.

# 5. IMPROVEMENT IN NUMERICAL OPTIMIZATION

## 5.1 Modified Optimization Technique

It is difficult to get a stable solution for the objective function using traditional least squares method if the matrix is ill-conditioned. Even with regularization, the solutions from the previous numerical methods are also different. This is because of the differences of the regularization methods. In this section, a new optimization method is introduced which was not used for this problem before. First, a vector of offset data is introduced in the fitting model. Second, a new numerical optimization method called Tikihonov [56] or Ridge Regression is used to find a relatively better solution, which is closer to the “best fit”. In a real experiment, there are different errors involved in measurement. These include random error (unbiased) and the systematic error (biased) from the experiment systems. Considering the influence of bias on the RR method, for example, the bias from background counts, or the bias from mathematical model of delayed neutron emission itself, the new objective function based on Eq. (4-24) is described to be:

$$\Delta\mathbf{N} = \mathbf{J}\Delta\mathbf{X} + \mathbf{R} \tag{5-1}$$

**R** is a vector of biased item, and **J** is Jacobian matrix of the fitted exponential model (see Eq. (4-26)). In this linear model, if matrix **J** is ill-conditioned, the solution will be very unstable or even not exist. As mentioned in Section 4, singular value decomposition is one way to deal with this problem. But singular value decomposition method can fail due to rank deficiency or round off error which will cause the residual

$|\mathbf{J}\Delta\mathbf{X}-\Delta\mathbf{N}|$ to be very large. In this case, it is essential to regularize the estimate of the regression coefficients by introducing a small amount of bias into the matrix, which can dramatically reduce the variance. There are several ways to accomplish this. The truncated singular value decomposition (TSVD) filters out both the zero values and the "small" values of $\omega_i$ that are relatively very small compared with a certain threshold. Another regularization method is Tikhonov regularization [56], which is also called Ridge Regression. Compared to TSVD, this method is better because Tikhonov used a very "smooth" regularization parameter to reduce the variance. It does not just truncate the small values of $\omega_i$.

According to the method of Tikhonov optimization, a Tikhonov regularization multiplier is introduced together with the constraint conditions to get the best fit of the parameters to minimize errors. Therefore, the new least squares equation is described as below:

$$\begin{aligned}&\sum_{i=1}^{m} w_i r_i^2 + h^2 \sum_{j=1}^{p} (\Delta x_j{}^2 - \Delta x_{j0}^2) + h^2\theta^2 \\ &= \sum_{i=1}^{m} w_i \left( \Delta N_i^0 - \sum_{j=1}^{p} J_{ij}^0 \Delta x_j - b_i^0 \theta \right)^2 + h^2 \sum_{j=1}^{p} (\Delta x_j{}^2 - \Delta x_{j0}^2) + h^2\theta^2\end{aligned} \tag{5-2}$$

Here, $h^2$ is a flexible regularization multiplier acting as a weighting factor to minimize the parameter deviation $\mathbf{R}=\mathbf{b}*\theta$ . Here, $\mathbf{b}$ is a vector of un-weighted constant, which is equivalent to the dwell times at different time periods, $b_i^0$ is an element of matrix b corresponding to the $i^{th}$ observation, and $\theta$ is a constant count rate due to the system error from the fitting model offset, etc. To find the optimal values of $\Delta x$ and $\theta$ to minimize the sum square residuals, set the partial derivatives of the above expression with respect to Δx and θ equal to zero.

The derivative to $\Delta x_k$ is:

$$\frac{\partial}{\partial \Delta x_k}\left\{\sum_{i=1}^{m} w_i\left(\Delta N_i^0 - \sum_{j=1}^{p} J_{ij}^0 \Delta x_j - b_i^0\theta\right)^2 + h^2\sum_{j=1}^{p}(\Delta x_j{}^2 - \Delta x_{j0}^2) + h^2\theta^2\right\} = 0$$

$$\Rightarrow \sum_{i=1}^{m}\left\{2w_i\left(\Delta N_i^0 - \sum_{j=1}^{p} J_{ij}^0 \Delta x_j - b_i^0\theta\right)(-J_{ik}^0)\right\} + 2h^2\Delta x_k = 0 \qquad (5\text{-}3)$$

$$\Rightarrow \sum_{i=1}^{m}\left\{\sqrt{w_i}\left(\Delta N_i^0 - \sum_{j=1}^{p} J_{ij}^0 \Delta x_j - b_i^0\theta\right)(\sqrt{w_i}J_{ik}^0)\right\} = h^2\Delta x_k$$

Apply this to all parameters, then

$$\mathbf{J^T\Delta N = (J^TJ + h^2I)\Delta x + \theta J^Tb} \qquad (5\text{-}4)$$

The derivative with respect to θ is:

$$\frac{\partial}{\partial \theta}\left\{\sum_{i=1}^{m} w_i\left(\Delta N_i^0 - \sum_{j=1}^{p} J_{ij}^0 \Delta x_j - b_i^0\theta\right)^2 + h^2\sum_{j=1}^{p}(\Delta x_j{}^2 - \Delta x_{j0}^2) + h^2\theta^2\right\} = 0$$

$$\Rightarrow \sum_{i=1}^{m} 2w_i\left(\Delta N_i^0 - \sum_{j=1}^{p} J_{ij}^0 \Delta x_j - b_i^0\theta\right)(-b_i^0) + 2h^2\theta = 0 \qquad (5\text{-}5)$$

Written in matrix form as:

$$\mathbf{b^T\Delta N = b^TJ\Delta x + \theta(b^Tb + h^2)} \qquad (5\text{-}6)$$

Set $\sqrt{w_i}J_{ik}^0 = J_{ik}$ , and $\sqrt{w_i}b_i^0 = b_i$ , the unknown parameters Δx and θ in two equations Eq. (5-4) and Eq. (5-6) can be written in matrix form as:

$$\begin{pmatrix}\mathbf{\Delta x}\\ \boldsymbol{\theta}\end{pmatrix} = \begin{pmatrix}\mathbf{J^TJ + h^2I} & \mathbf{J^Tb}\\ \mathbf{b^TJ} & \mathbf{b^Tb + h^2}\end{pmatrix}^{-1}\begin{bmatrix}\mathbf{J^T\Delta N}\\ \mathbf{b^T\Delta N}\end{bmatrix} \qquad (5\text{-}7)$$

$\mathbf{J}^T$ is the transpose matrix of **J**, and **I**(p×p) is an identity matrix, θ is a constant. Similar to Levenberg-Marquardt expression, the expression in Eq. (5-7) can be easily changed into a similar form:

$$\mathbf{X^*} = \begin{pmatrix} \mathbf{\Delta x} \\ \boldsymbol{\theta} \end{pmatrix} = \left( \left[ \mathbf{J} \quad \mathbf{b} \right]^T \left[ \mathbf{J} \quad \mathbf{b} \right] + \mathbf{h^2 I} \right)^{-1} \left[ \mathbf{J} \quad \mathbf{b} \right]^T \mathbf{\Delta N} \qquad (5\text{-}8)$$

Here, **b** is an m×1 known vector, and now **I** is a (p+1) × (p+1) matrix in this new expression. The definition of the symbols is: $\sqrt{w_i} J_{ij}^0 = J_{ij}$, $\sqrt{w_i} b_i^0 = b_i$, $\sqrt{w_i} \Delta N_i^0 = \Delta N_i$. Matrix **J** is the original Jacobian matrix from the model function. When **b**=0, the above equation is just exactly the same as Levenberg-Marquardt algorithm.

For simplicity, the solution of the above equation is written as:

$$\mathbf{X^*} = (\mathbf{A}^T \mathbf{A} + h^2 \mathbf{I})^{-1} \mathbf{A}^T \mathbf{B} \qquad (5\text{-}9)$$

Matrix **A** (m×p) is the design matrix (in this equation, **A**=[**J b**], **B** (p×1) is the observation vector with **B**=Δ**N**, and h is the regularization parameter whose role is to determine the trade off between minimizing the residual sum squares and minimizing the norm of the parameter.

For any nonzero value h, the matrix $(\mathbf{A}^T \mathbf{A} + h^2 \mathbf{I})$ will no longer be singular even if the matrix $\mathbf{A}^T \mathbf{A}$ is severely ill posed. If the value h is too small, the above solution will be still unstable; if h is too large, the item $(\mathbf{A}^T \mathbf{A} + h^2 \mathbf{I})$ will be a diagonal dominated matrix and the solution is dominated by initial input.

Applying the SVD method $\mathbf{A} = \mathbf{U} \mathbf{W} \mathbf{V}^T$ into the above equation:

$$\begin{aligned}\mathbf{X}^* &= ((\mathbf{UWV}^{T})^{T}\mathbf{UWV}^{T} + h^{2}\mathbf{I})^{-1}(\mathbf{UWV}^{T})^{T}\mathbf{B} \\ &= (\mathbf{VW}^{2}\mathbf{V}^{T} + \mathbf{V}(h^{2}\mathbf{I})\mathbf{V}^{T})^{-1}\mathbf{VWU}^{T}\mathbf{B} \\ &= (\mathbf{V}\mathrm{diag}(\omega_i^2+h^2)\mathbf{V}^{T})^{-1}\mathbf{V}\mathrm{diag}(\omega_i)\mathbf{U}^{T}\mathbf{B} \\ &= \mathbf{V}\mathrm{diag}(\frac{\omega_i}{\omega_i^2+h^2})\mathbf{U}^{T}\mathbf{B} \\ &= \sum_{i=1}^{p} f_i \frac{\mathbf{u_i^T B}}{\omega_i}\mathbf{v_i}\end{aligned} \qquad (5\text{-}10)$$

Here, the filter factor $f_i$ is defined as

$$f_i = \frac{\omega_i^2}{\omega_i^2+h^2} \qquad (5\text{-}11)$$

If $\omega_i >> h$, then $f_i \approx 1$, the right singular vectors $v_i$ is retained; if $\omega_i << h$, then $f_i \approx 0$, the right singular vector element $v_i$ is filtered out due to the small ratio of "signal" ($\omega_i^2$) to "noise" ($h^2$). TSVD is similar to the above smooth filter factor with:

$$f_i = \begin{cases} 1 & \text{for } i \leq \text{ effective rank} \\ 0 & \text{for } i > \text{effective rank} \end{cases} \qquad (5\text{-}12)$$

In TSVD, the regularization parameter is discrete; while in Ridge Regression, it is continuous that it might suppress noise with less loss of relevant information than TSVD.

For small levels of measurement errors, the relationship between the random errors in measurements and the resulting random errors in parameter values can be considered linear and the covariance of the estimated parameter is similar to that of SVD covariance in the following expression structure:

$$\begin{aligned}\mathbf{C_X} &= (\mathbf{A^TA}+h^2\mathbf{I})^{-1}\mathbf{A^TC_bA}(\mathbf{A^TA}+h^2\mathbf{I})^{-1}\\ &= \left[(\mathbf{UWV}^T)^T\mathbf{UWV}^T+h^2\mathbf{I}\right]^{-1}(\mathbf{UWV}^T)^T*\mathbf{C}_b*\\ &\quad(\mathbf{UWV}^T)\left[(\mathbf{UWV}^T)^T\mathbf{UWV}^T+h^2\mathbf{I}\right]^{-1}\\ &= \left[\mathbf{VW}^2\mathbf{V}^T+\mathbf{V}(h^2\mathbf{I})\mathbf{V}^T\right]^{-1}(\mathbf{VWU}^T)*\mathbf{C}_b*\\ &\quad(\mathbf{UWV}^T)\left[\mathbf{VW}^2\mathbf{V}^T+\mathbf{V}(h^2\mathbf{I})\mathbf{V}^T\right]^{-1}\\ &= \left[\mathbf{V}\mathrm{diag}(\omega_i^2+h^2)\mathbf{V}^T\right]^{-1}\mathbf{VWU}^T*\mathbf{C}_b*\\ &\quad\mathbf{UWV}^T\left[\mathbf{V}\mathrm{diag}(\omega_i^2+h^2)\mathbf{V}^T\right]^{-1}\\ &= \sigma^2\left[\mathrm{diag}\left(\frac{\omega_i}{\omega_i^2+h^2}\right)\right]^2\mathbf{VV^T} \qquad (\text{suppose } \mathbf{C}_b=\sigma^2\mathbf{I})\end{aligned} \tag{5-13}$$

$\mathbf{C_X}$ is the covariance matrix of the unknown parameters and $\mathbf{C}_b$ is the covariance matrix of measurement error which is assumed to be white noise and equal to $\sigma^2\mathbf{I}$. $\sigma^2$ is standard deviation of the observations, which is the value of sum of weighted squared residuals divided by degree number. Thus, the covariance matrix element between parameter $x_j$ and $x_k$ is derived as:

$$\mathrm{Cov}(x_j,x_k)=\sigma^2\sum_{i=1}^{p}\left(\frac{\omega_i^2}{\omega_i^2+h^2}\right)^2\left(\frac{V_{ji}V_{ki}}{\omega_i^2}\right) \tag{5-14}$$

The above expression is similar to that of the result from SVD method except for the additional item of the filter factor square as a multiplier.

For the least squares method, there is no bias, but for the Ridge Regression method, due to perturbation from the regularization parameter, there is bias. The mean square error, MSE, is just the mean squared difference between the fitted estimation $\hat{\mathbf{X}}*$ and the expected values $\mathbf{X}$ from Ridge Regression [57]. The MSE is explained through the derivation of the following expressions.

$$\begin{aligned}\hat{\mathbf{X}}^{*} &= [\mathbf{A}^{\mathbf{T}}\mathbf{A}+\mathrm{h}^{2}\mathbf{I}]^{-1}\mathbf{A}^{\mathbf{T}}\mathbf{B}\\ &= [\mathbf{I}+\mathrm{h}^{2}(\mathbf{A}^{\mathbf{T}}\mathbf{A})^{-1}]^{-1}(\mathbf{A}^{\mathbf{T}}\mathbf{A})^{-1}\mathbf{A}^{\mathbf{T}}\mathbf{B}\\ &= [\mathbf{I}+\mathrm{h}^{2}(\mathbf{A}^{\mathbf{T}}\mathbf{A})^{-1}]^{-1}\hat{\mathbf{X}}\\ &= \mathbf{Z}\hat{\mathbf{X}}\end{aligned} \tag{5-15}$$

Here, $\hat{\mathbf{X}}$ is the solution vector from unbiased ordinary least squares method. Z is set to be $[\mathbf{I}+\mathrm{h}^{2}(\mathbf{A}^{\mathbf{T}}\mathbf{A})^{-1}]^{-1}$ and can be further derived from

$$\begin{aligned} & [\mathbf{A}^{\mathbf{T}}\mathbf{A}+\mathrm{h}^{2}\mathbf{I}]^{-1}\mathbf{A}^{\mathbf{T}}\mathbf{A} = [\mathbf{A}^{\mathbf{T}}\mathbf{A}+\mathrm{h}^{2}\mathbf{I}]^{-1}[\mathbf{A}^{\mathbf{T}}\mathbf{A}+\mathrm{h}^{2}\mathbf{I}-\mathrm{h}^{2}\mathbf{I}]\\ \Rightarrow\ & [\mathbf{I}+\mathrm{h}^{2}(\mathbf{A}^{\mathbf{T}}\mathbf{A})^{-1}]^{-1} = \mathbf{I}-\mathrm{h}^{2}[\mathbf{A}^{\mathbf{T}}\mathbf{A}+\mathrm{h}^{2}\mathbf{I}]^{-1}\\ \Rightarrow\ & \mathbf{Z} = \mathbf{I}-\mathrm{h}^{2}[\mathbf{A}^{\mathbf{T}}\mathbf{A}+\mathrm{h}^{2}\mathbf{I}]^{-1}\end{aligned} \tag{5-16}$$

The mean squared error can be derived as:

$$\begin{aligned}\mathrm{MSE} &= \mathrm{E}[(\hat{\mathbf{X}}^{*}-\mathbf{X})^{\mathrm{T}}(\hat{\mathbf{X}}^{*}-\mathbf{X})]\\ &= \mathrm{E}[(\hat{\mathbf{X}}-\mathbf{X})^{\mathrm{T}}\mathbf{Z}^{\mathrm{T}}\mathbf{Z}(\hat{\mathbf{X}}-\mathbf{X})]+(\mathbf{Z}\mathbf{X}-\mathbf{X})^{\mathrm{T}}(\mathbf{Z}\mathbf{X}-\mathbf{X})\\ &= \sigma^{2}\mathrm{Trace}(\mathbf{A}^{\mathrm{T}}\mathbf{A})^{-1}\mathbf{Z}^{\mathrm{T}}\mathbf{Z}+\mathbf{X}^{\mathrm{T}}(\mathbf{Z}-\mathbf{I})^{\mathrm{T}}(\mathbf{Z}-\mathbf{I})\mathbf{X}\\ &= \sigma^{2}\left\{\mathrm{Trace}(\mathbf{A}^{\mathrm{T}}\mathbf{A}+\mathrm{h}^{2}\mathbf{I})^{-1}-\mathrm{h}^{2}\mathrm{Trace}(\mathbf{A}^{\mathrm{T}}\mathbf{A}+\mathrm{h}^{2}\mathbf{I})^{-2}\right\}\\ &\quad +\mathrm{h}^{4}\mathbf{X}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathbf{A}+\mathrm{h}^{2}\mathbf{I})^{-2}\mathbf{X}\\ &= \underline{\underline{\sigma^{2}[(\mathbf{A}^{\mathrm{T}}\mathbf{A}+\mathrm{h}^{2}\mathbf{I})^{-2}\mathbf{A}^{\mathrm{T}}\mathbf{A}]}}+\underline{\mathrm{h}^{4}\mathbf{X}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathbf{A}+\mathrm{h}^{2}\mathbf{I})^{-2}\mathbf{X}}\\ &= \underline{\underline{\sigma^{2}\sum_{i=1}^{p}\frac{\lambda_{i}}{(\lambda_{i}+\mathrm{h}^{2})^{2}}}}+\underline{\mathrm{h}^{4}\mathbf{X}^{\mathrm{T}}(\mathbf{A}^{\mathrm{T}}\mathbf{A}+\mathrm{h}^{2}\mathbf{I})^{-2}\mathbf{X}}\\ &= \underline{\underline{\mathrm{Var}(\hat{\mathbf{X}}^{*})}}+\underline{[\mathrm{bias}(\hat{\mathbf{X}}^{*})]^{2}}\end{aligned} \tag{5-17}$$

Here, **X** is replaced by $\hat{\mathbf{X}}^{*}$ for the bias approximation calculation. In Eq. (5-17), $\lambda_i$ $(\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_p > 0)$ is the eigenvalue of matrix $\mathbf{A}^{\mathbf{T}}\mathbf{A}$, and $\omega_i$ $(\omega_1 \geq \omega_2 \geq \ldots \geq \omega_p > 0)$ is the singular value of matrix **A** mentioned before. Trace is the sum of diagonal entries of a matrix. From the above expression, it is obvious that Ridge Regression reduces the variance dramatically, especially for a matrix with very small eigenvalues. However, on the other hand, it also introduces bias. Increasing the regularization parameter h will increase the bias. Figure 5.1 (This figure is obtained from Hoel and Kennard [57]) shows in qualitative form the relationship between variance, bias, and the regularization parameter. The above equation for variance and bias of the model parameters can be expressed as:

$$\mathrm{Var}\left[(\hat{\mathbf{X}}^{*})_{\mathrm{i}}\right]=\sigma^{2}(\mathbf{A}^{*}*\mathbf{A}^{*\mathrm{T}})_{\mathrm{ii}}=\sigma^{2}\sum_{\mathrm{i=1}}^{\mathrm{p}}\frac{\lambda_{\mathrm{i}}}{(\lambda_{\mathrm{i}}+\mathrm{h}^{2})^{2}} \qquad (5\text{-}18)$$

$$\mathbf{A}^{*}=(\mathbf{A}^{\mathrm{T}}\mathbf{A}+\mathrm{h}^{2}\mathbf{I})^{-1}\mathbf{A}^{\mathrm{T}} \qquad \mathrm{Bias}(\hat{\mathbf{X}}^{*})=\mathrm{E}(\hat{\mathbf{X}}^{*}-\mathbf{X})=-\mathrm{h}^{2}(\mathbf{A}^{\mathrm{T}}\mathbf{A}+\mathrm{h}^{2}\mathbf{I})^{-1}\mathbf{X} \qquad (5\text{-}19)$$

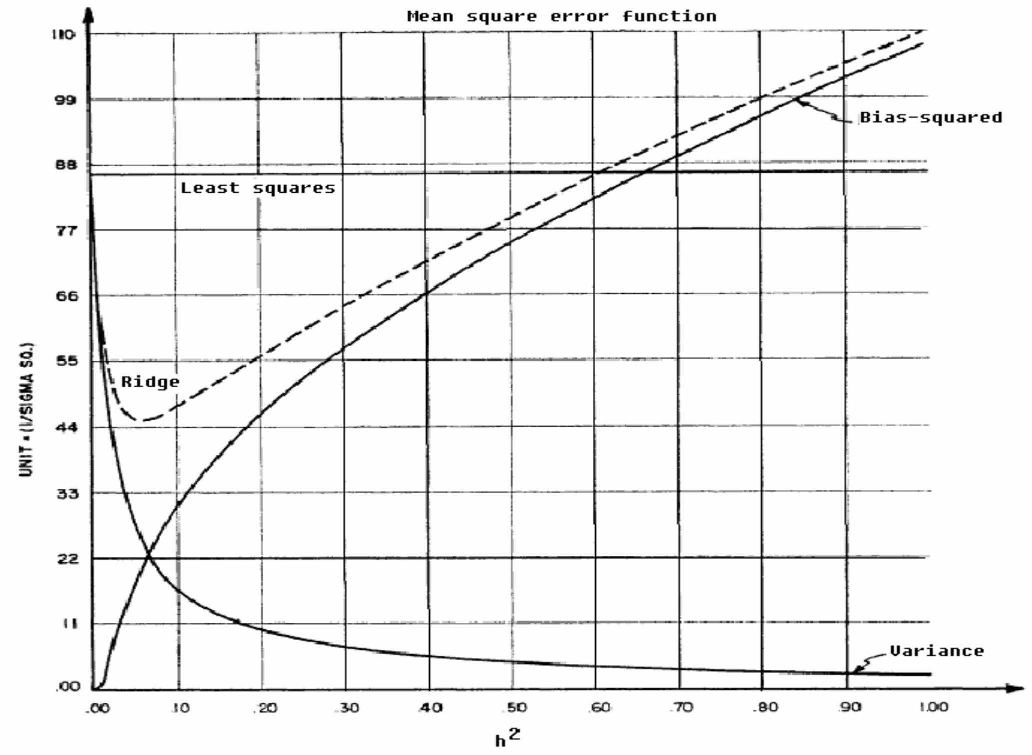


Figure 5.1 Comparisons of Mean Square Error Functions between Least Squares and Ridge Regression Method [57]

## 5.2 Methods to Find the Regularization Parameter

The best way to minimize MSE from Ridge Regression is to find a trade-off parameter h to balance these two errors. A reasonable regularization parameter should satisfy this goal

$$\frac{\mathrm{RSS}}{\mathrm{m}}=\frac{||\mathbf{AX-B}||^2}{\mathrm{m}}\approx\sigma^2 \tag{5-20}$$

There are several ways to optimize h value. Two of them will be introduced here. The first method is to plot an L-curve to find the turning point, which was first proposed by Hansen [58]. The second is called the generalized cross-validation (GCV) [59].

### 5.2.1 L-curve

Hansen [58] suggested selecting h by inspecting the plot of the squared solution parameter norm $||\mathbf{X}||^2$ with respect to the residual sum of squares $||\mathbf{AX\text{-}B}||^2$ using a log-log scale. The items of $||\mathbf{X}||^2$ and $||\mathbf{AX\text{-}B}||^2$ [60] can be expressed as:

$$||\mathbf{X}||^2=\sum_{\mathrm{i=1}}^{\mathrm{p}}\left(\frac{\omega_{\mathrm{i}}^2}{\omega_{\mathrm{i}}^2+\mathrm{h}^2}\right)^2\left(\frac{\mathbf{u}_{\mathbf{i}}^{\mathbf{T}}\mathbf{B}}{\omega_{\mathrm{i}}}\right)^2 \tag{5-21}$$

$$||\mathbf{AX-B}||^2=\sum_{\mathrm{i=1}}^{\mathrm{m}}\mathbf{B}_{\mathrm{i}}^2-\sum_{\mathrm{i=1}}^{\mathrm{p}}\frac{\omega_{\mathrm{i}}^4(\omega_{\mathrm{i}}^2+2\mathrm{h}^2)}{(\omega_{\mathrm{i}}^2+\mathrm{h}^2)^2}\left(\frac{\mathbf{u}_{\mathbf{i}}^{\mathbf{T}}\mathbf{B}}{\omega_{\mathrm{i}}}\right)^2 \tag{5-22}$$

The shape of the curve is usually like “L”. The optimal value of h is just at the corner of L-curve, which separates the vertical and horizontal part of the curve (see Figure 5.2). The regularization parameter at the corner of the curve is h ≈ 1.0. When the regularization parameter h is smaller than the corner value, the squared solution parameter norm $||\mathbf{X}||^2$ will increase greatly with almost the same value of the residual sum of squares $||\mathbf{AX}\text{-}\mathbf{B}||^2$. This means that many values can fit the objective function very well at almost the same sum of weighted squared residuals point while with very large estimator variation. The solution is thus very unstable in this region. The code to plot L-curve is given in MATLAB code in Appendix A.5.

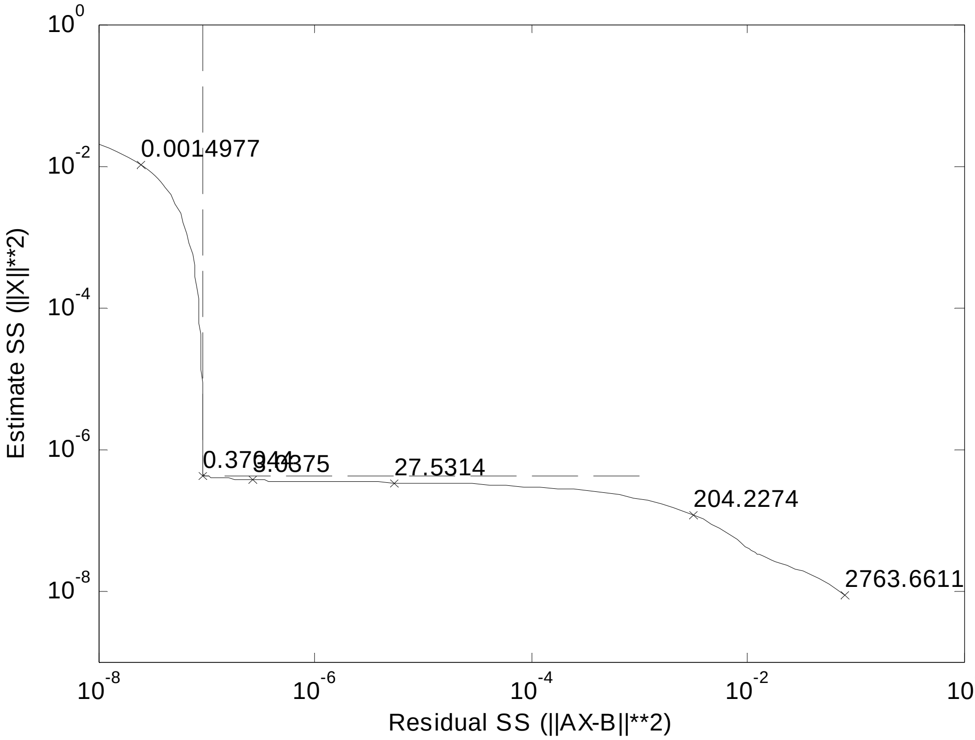


Figure 5.2 L-curve for Decay Yield from Matrix Inverse Method

### 5.2.2 Generalized Cross-validation

Generalized cross-validation uses leave-one-out method to remove elements $b_i$ from the observation vector **B**, compute a regularized parameter vector **X** from the reduced data set, and then predict the left-out observation $b_i$ with the model [61]. The optimal regularization parameter h is selected by optimizing the GCV function:

$$\text{GCV}=\frac{\text{RSS}}{\tilde{r}^2} \tag{5-23}$$

where

$$\tilde{r}=m-\sum_{i=1}^{p} f_i = m-\sum_{i=1}^{p}\frac{\omega_i^2}{\omega_i^2+h^2} \tag{5-24}$$

Here, RSS is the residual sum of squares, $\tilde{r}$ is the effective number of degrees of freedom and $f_i$ is the filter factor in Tikhonov regularization method. $\tilde{r}$ may not be an integer. The total sum of residual squares RSS is partitioned into two parts: a sum of squares due to regression (biased) (RSSR) and a sum of squares due to error (unbiased) (RSSE). That is, RSS=RSSR+RSSE. Using SVD method, it can be expressed as:

$$\text{RSS}=\left\|\mathbf{B}\text{-}\sum_{i=1}^{p}(\mathbf{u}_i^T\mathbf{B})\mathbf{u}_i\right\|^2+\left\|\sum_{i=1}^{p}\left(\frac{h^2}{\omega_i^2+h^2}\right)(\mathbf{u}_i^T\mathbf{B})\mathbf{u}_i\right\|^2 \tag{5-25}$$

Increasing the regularization parameter h will increase the degree of freedom and usually also increase the residual of sum of squares. The optimal regularization value h is at the minimization point of GCV (see Figure 5.3). The GCV program code is written in MATLAB in Appendix A.5. Compared to the value from L-curve in Figure 5.2, the h value from GCV is very close to that from L-curve.

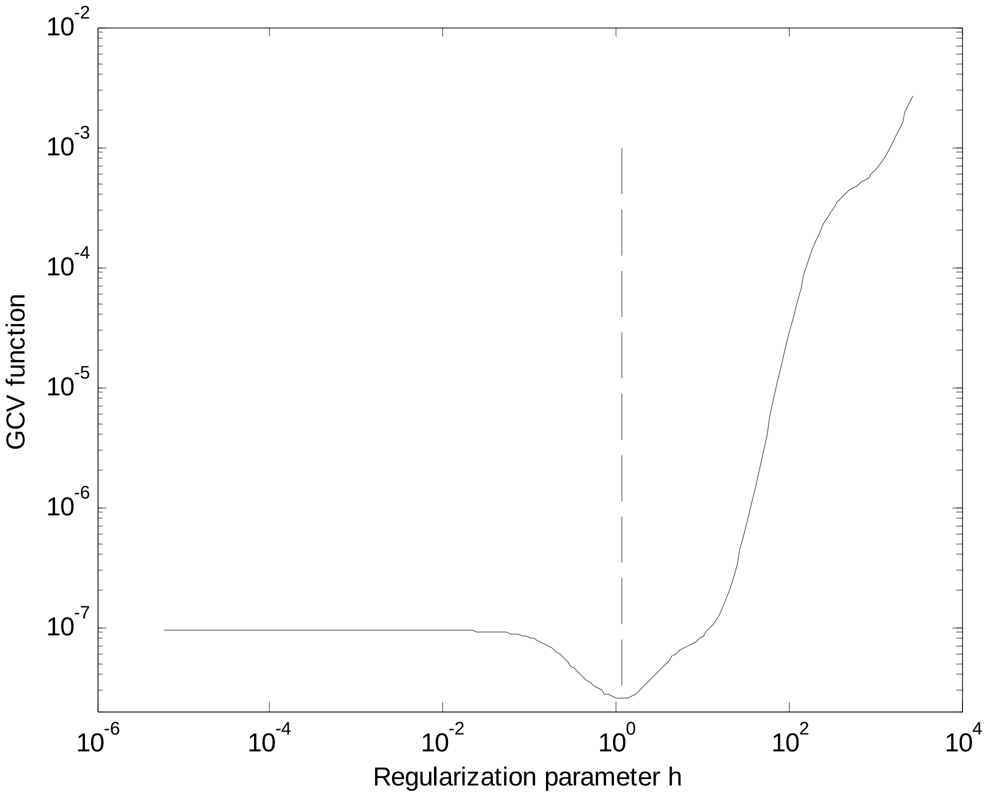


Figure 5.3 GCV Function for Decay Yield from Matrix Inverse Method

# 6. RESULTS AND DISCUSSION

Based on the theories mentioned previously, this section illustrates the calculated results using the Ridge Regression (RR) method under different conditions. The fitted results from RR are compared to the results from other numerical methods, such as MI-SVD, LM and QN. Optimization results from different mathematical models are compared, such as the results obtained by freezing one or more of the parameters. Both saturation and pulse modes are examined, and the factors affecting the fitted results are analyzed. These include the sample flight time, initial count rate, and initial guessed values. To check the goodness of fit, the fitted data sets are compared to Keepin's values, which are used to generate the input data. Finally, real experimental data sets are applied to verify the simulated results. The experiments were done in the TRIGA reactor at the Nuclear Science Center at Texas A&M University [62]. The data sets are listed in Appendix C.

## 6.1 Results from the Ridge Regression Algorithm

Ridge Regression can always converge if the regularization parameters are properly selected. The same data sets are used for the calculation. The fitted results from RR are listed below in Table 6.1 (saturation mode) and Table 6.2 (pulse mode). All unknown parameters are variable. The values $A_i/A_i'$ ($\lambda_i/\lambda_i'$) listed in the tables are the ratios of the fitted relative yield $A_i$ (decay constant $\lambda_i$) to the Keepin's values $A_i'$ ($\lambda_i'$). $E^2$ is the sum of weighted squared differences between the fitted counts and

Keepin's theoretical counts at different decay times. The selection of parameter h for each data set is based on the "L-Curve" method. The initial count rate in all the simulated data sets is assumed to be Poisson distributed around the same value (4.0E+5 cps as an example). The sample has zero transfer flight time.

The extracted ratios from RR also cannot return to 1.0. In fact, some are far from 1.0. This indicates that the extracted results from RR are unstable. Compared to the extracted results from other methods, such as MI-SVD (Table 4.2), LM (Table 4.3), QN (Table 4.4), this method is relatively better. RR can always converge and find a solution. The fitted $E^2$ values using RR are the smallest among these algorithms (see Figure 6.1). QN can achieve very small $E^2$ values, close to RR values in most of the cases, but the QN method cannot guarantee convergence for some data sets (see Table 4.4). That means RR achieves better optimization results.

The Matrix Inverse method is derived from the least squares method. Its solution is relatively more stable than other algorithms if the coefficient matrix is not ill-posed. For the ill-posed problem, the Matrix Inverse method with SVD always leads to a solution. However, the fitted result may have very large $E^2$ values. LM attains a better solution than MI-SVD, but its $E^2$ is generally not the smallest. Compared to other algorithms, RR gives relatively more stable fitted values for longer-lived groups in saturation mode, such as Group-1, Group-2 and Group-3. For shorter-lived groups, it has larger variation. In pulse mode, the variations for short-lived groups decrease, but for long-lived groups, the variations increase compared to that in saturation mode (see Group-1 in Table 6.2).

Table 6.1 Fitted Results from Ridge Regression (Saturation Mode)

| seed | -6295779 | -389279151 | 738164549 | -1778841095 | 1234 | -1031501663 | -1141185387 | 774385225 | 397015485 | 534354673 | 4250 | -25 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| h | 0.6 | 0.4 | 0 | 0.1 | 0.7 | 0.9 | 0.5 | 0.15 | 0.4 | 0.8 | 0.35 | 0.4 |
| $A_1/A_1'$ | 0.8863 | 1.0608 | 0.7054 | 1.0060 | 0.9776 | 1.0669 | 0.9902 | 0.9628 | 1.0304 | 0.8964 | 1.0013 | 0.9746 |
| $A_2/A_2'$ | 0.9572 | 1.0126 | 0.9976 | 0.9914 | 1.0084 | 0.9896 | 0.9976 | 1.0038 | 0.9925 | 0.8353 | 1.0105 | 0.9972 |
| $A_3/A_3'$ | 0.5500 | 1.1436 | 0.9529 | 0.9374 | 1.0024 | 0.8900 | 1.0149 | 0.9038 | 0.8482 | 1.1587 | 1.0718 | 1.0823 |
| $A_4/A_4'$ | 0.5972 | 0.9654 | 1.0341 | 1.0077 | 0.9798 | 0.9255 | 0.9877 | 0.8496 | 1.0232 | 0.9388 | 0.9926 | 1.0304 |
| $A_5/A_5'$ | 2.4703 | 0.2464 | 0.9851 | 1.1673 | 0.8550 | 0.9945 | 0.9823 | 1.2851 | 1.2803 | 1.1461 | 0.3713 | 1.0640 |
| $A_6/A_6'$ | 3.1784 | 2.7052 | 1.3387 | 0.9398 | 1.3750 | 2.1534 | 1.1236 | 1.9600 | 0.8525 | 1.1902 | 2.3224 | 1.6314 |
| $\lambda_1/\lambda_1'$ | 0.9679 | 1.0160 | 0.8515 | 1.0050 | 0.9927 | 1.0212 | 0.9984 | 0.9877 | 1.0120 | 0.9783 | 0.9999 | 0.9932 |
| $\lambda_2/\lambda_2'$ | 0.9637 | 1.0160 | 0.9500 | 0.9972 | 0.9992 | 1.0080 | 0.9966 | 0.9954 | 1.0030 | 1.0935 | 1.0036 | 0.9935 |
| $\lambda_3/\lambda_3'$ | 0.7307 | 1.0874 | 0.9086 | 0.9581 | 1.0101 | 0.9551 | 0.9982 | 0.9693 | 0.9313 | 1.1310 | 1.0504 | 1.0100 |
| $\lambda_4/\lambda_4'$ | 0.6110 | 1.0929 | 0.9699 | 0.9615 | 1.0002 | 0.9127 | 0.9998 | 0.8740 | 0.9180 | 1.1022 | 1.0446 | 1.0827 |
| $\lambda_5/\lambda_5'$ | 0.3220 | 0.7373 | 0.9112 | 0.9485 | 0.8194 | 1.4618 | 0.9712 | 0.5385 | 0.9122 | 1.3716 | 0.9113 | 1.4195 |
| $\lambda_6/\lambda_6'$ | 0.5441 | 0.7221 | 1.0205 | 1.7972 | 0.7760 | 0.1827 | 0.9245 | 0.6993 | 1.5559 | 0.0079 | 0.6341 | 15.8420 |
| $E^2$ | 640.52 | 660.08 | 560.22 | 651.29 | 666.87 | 605.61 | 567.79 | 574.30 | 621.35 | 658.42 | 590.84 | 583.96 |

Note: 1) Ai/Ai' ( λ i/ λ i') are the ratios of DN abundance and decay constant to Keepin’s values for group i. 2) E2 is the sum of weighted squared residuals between observation and fitted count. 3) h is the regularization parameter. They are the same in the following tables.

Table 6.2 Fitted Results from Ridge Regression with No Parameter Fixed (Saturation Mode)

| seed | -6295779 | -389279151 | 738164549 | -1778841095 | 1234 | -1031501663 | -1141185387 | 774385225 | 397015485 | 534354673 |
|---|---|---|---|---|---|---|---|---|---|---|
| h | 0 | 0.8 | 0.2 | 0 | 0.2 | 0 | 0.1 | 0 | 0.2 | 0 |
| $A_1/A_1'$ | 0.6782 | 1.3175 | 0.9142 | 3.5766 | 0.8199 | 1.2048 | 0.7682 | 0.8183 | 0.8482 | 2.1571 |
| $A_2/A_2'$ | 0.9854 | 0.9947 | 1.0072 | 0.6896 | 1.0429 | 0.9485 | 1.0366 | 1.0264 | 1.0026 | 0.8879 |
| $A_3/A_3'$ | 0.8977 | 1.0602 | 1.0461 | 0.9962 | 1.0048 | 0.9261 | 1.0448 | 0.9472 | 0.9272 | 0.9762 |
| $A_4/A_4'$ | 1.0334 | 0.9456 | 0.9887 | 0.9693 | 0.9839 | 1.0190 | 0.9885 | 0.9945 | 1.0450 | 0.9371 |
| $A_5/A_5'$ | 0.9604 | 0.9027 | 0.9953 | 1.0521 | 0.9889 | 0.9562 | 0.9988 | 1.0553 | 1.0782 | 0.8793 |
| $A_6/A_6'$ | 1.5876 | 1.2573 | 0.9513 | 0.7086 | 1.0872 | 1.4263 | 0.9493 | 1.1619 | 0.8548 | 1.6196 |
| $\lambda_1/\lambda_1'$ | 0.8936 | 1.0992 | 0.8723 | 1.6594 | 0.8334 | 1.0837 | 0.7666 | 0.8618 | 0.8654 | 1.3645 |
| $\lambda_2/\lambda_2'$ | 0.9419 | 1.0499 | 0.9870 | 1.2874 | 0.9912 | 1.0041 | 0.9727 | 0.9853 | 0.9770 | 1.1412 |
| $\lambda_3/\lambda_3'$ | 0.8699 | 1.0845 | 1.0063 | 1.1035 | 1.0125 | 0.9315 | 1.0117 | 0.9735 | 0.9217 | 1.0821 |
| $\lambda_4/\lambda_4'$ | 0.9267 | 1.0277 | 1.0198 | 1.0314 | 1.0039 | 0.9585 | 1.0120 | 0.9582 | 0.9696 | 0.9882 |
| $\lambda_5/\lambda_5'$ | 0.7941 | 0.9252 | 1.0247 | 1.0945 | 0.9553 | 0.8479 | 1.0272 | 0.8963 | 1.0226 | 0.8059 |
| $\lambda_6/\lambda_6'$ | 0.8284 | 0.9372 | 1.0351 | 1.1755 | 0.9557 | 0.8573 | 1.0189 | 0.9431 | 1.0779 | 0.8101 |
| $E^2$ | 642.07 | 660.18 | 564.82 | 649.54 | 666.04 | 604.25 | 568.37 | 577.07 | 619.25 | 656.17 |

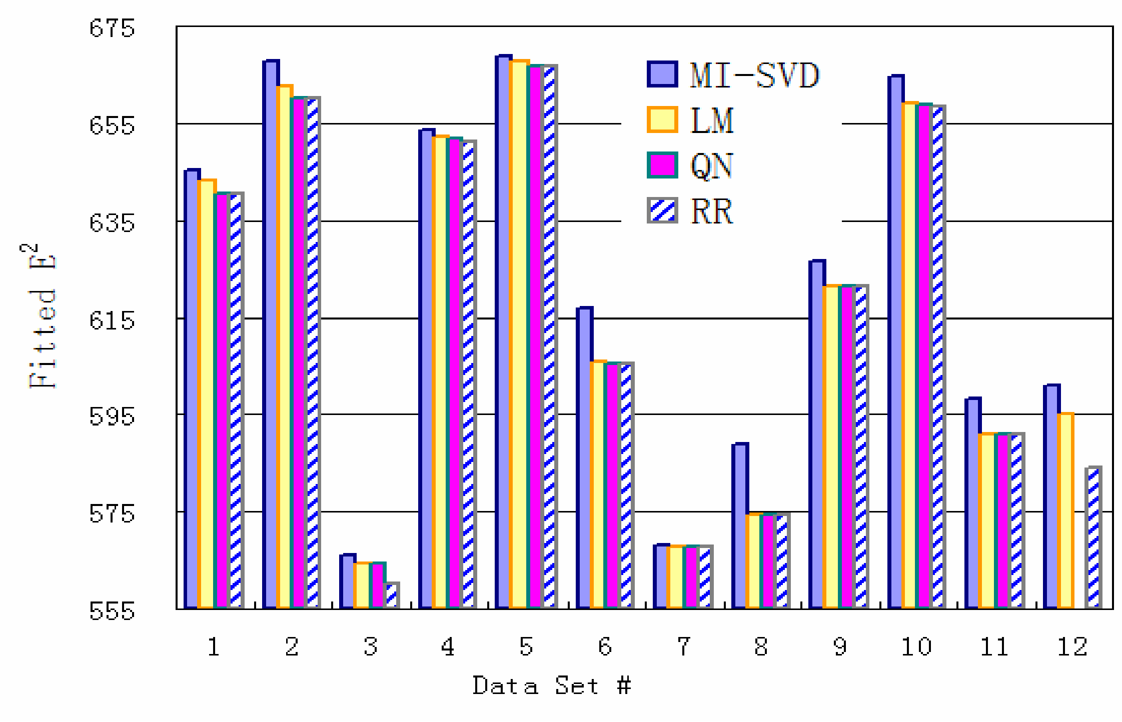


Figure 6.1 Fitted $E^2$ from Different Algorithms (Saturation Mode)
Note: No solution from QN algorithm for data set 12#.

To ensure the program codes are correct, a simple nonlinear least squares equation is used as an example. The calculated results from the different algorithms are listed in Appendix B, and the codes are listed in Appendices B.1, B.2, B.3, and B.4. From the results, it is clear that they can all obtain the correct solution to the problem. The codes and the methods have no apparent shortcomings. The function value F from RR is nonzero due to the machine precision. But it can be taken as zero due to its extremely small value.

In Table 6.1 and 6.2, all the unknown parameters are variable in the model. To attain a better optimization result, some parameters are fixed and the optimization results are compared. In the simulation, ten random numbers are chosen to get ten sets of results (same random seeds as those in Table 6.2). All of the data sets are generated from the same input parameters so that they have the same initial count rate before being randomized. To simplify the computation complexity, the six decay constants in the model are fixed, and the model becomes linear. The new extracted results for relative yields ratios are listed in Table 6.3 (saturation mode) and Table 6.4 (pulse mode). In this case, only seven unknown parameters exist in the objective function. Compared to the results in Table 6.1 (thirteen variables), this new fit provides a relatively more stable solution. The fitted ratio values are closer to 1.0. However, the $E^2$ values are larger, meaning that the overall fit quality is worse. Table 6.5 shows the results when the longest-lived group decay constant is fixed (twelve unknown group parameters left) in saturation mode. Table 6.6 shows the results when Group-1 parameters (yield and decay constant) are fixed in pulse mode. The reason to choose Group-1 is that it is very dominant in the later period of decay due to its long half-life and the fitted result is relatively more stable in saturation irradiation. From the results shown in the tables, it is obvious that when all unknown parameters are variable, the fitted results are the best because $E^2$ values are the smallest. The smaller the number of the fitted parameters, the worse the optimization result. This conclusion is similar to that in Geslot et al. [48]. If all parameters are variable, smaller $E^2$ values are achieved, but the fitted parameters have larger uncertainty compared to these values with part of the parameters fixed, as reflected in Tables 6.3 and 6.4.

The fitted results from the six-group model obtained by Geslot et al. [48] are listed in Table 2.8. Their fitted results are closer to Tuttle's values (Note: Tuttle's values are the same as Keepin's values), but the fitted results from RR here have larger discrepancy from Keepin's values. This is because Geslot used only a single fit based on 140 precursors in the JEFF-3.1 data library. There was no statistical variation in their merged data set. Another reason may be their numerical method, which was not mentioned in their paper. Other researchers, such as Loaiza [63], have suggested using fixed decay constants in the model due to its computational simplicity. The results in this study suggest that this model gives a worse fit and it is not recommended here. The most variable results are from short decay groups, such as Group-6. This conclusion is the same as that made by Geslot et al. [48] in their results.

Table 6.3 Fitted Results from Ridge Regression with Six Decay Constants Fixed (Saturation Mode)

| seed | -6295779 | -389279151 | 738164549 | -1778841095 | 1234 | -1031501663 | -1141185387 | 774385225 | 397015485 | 534354673 |
|---|---|---|---|---|---|---|---|---|---|---|
| $A_1/A_1'$ | 0.9896 | 0.9859 | 0.9936 | 1.0010 | 1.0373 | 1.0213 | 0.9946 | 1.0284 | 0.9941 | 1.0175 |
| $A_2/A_2'$ | 0.9925 | 0.9841 | 0.9900 | 1.0018 | 1.0442 | 1.0204 | 0.9965 | 1.0307 | 1.0037 | 1.0116 |
| $A_3/A_3'$ | 0.9936 | 0.9821 | 0.9959 | 1.0057 | 1.0297 | 1.0176 | 1.0022 | 1.0386 | 0.9935 | 1.0234 |
| $A_4/A_4'$ | 0.9886 | 0.9859 | 0.9874 | 0.9986 | 1.0434 | 1.0258 | 0.9891 | 1.0179 | 1.0065 | 0.9927 |
| $A_5/A_5'$ | 0.9974 | 0.8556 | 1.0131 | 0.9913 | 1.1015 | 1.0607 | 1.0101 | 1.1119 | 0.9975 | 1.0890 |
| $A_6/A_6'$ | 1.0752 | 1.4358 | 1.0710 | 1.0002 | 0.6954 | 0.8327 | 1.0115 | 0.7396 | 1.0043 | 0.8065 |
| $E^2$ | 646.17 | 663.44 | 566.45 | 654.19 | 668.94 | 612.66 | 568.57 | 578.21 | 626.36 | 663.26 |

Table 6.4 Fitted Results from Ridge Regression with Six Decay Constants Fixed (Pulse Mode)

| seed | -6295779 | -389279151 | 738164549 | -1778841095 | 1234 | -1031501663 | -1141185387 | 774385225 | 397015485 | 534354673 |
|---|---|---|---|---|---|---|---|---|---|---|
| $A_1/A_1'$ | 0.9811 | 1.0041 | 1.0061 | 0.9953 | 0.9818 | 0.9985 | 0.9880 | 0.9854 | 0.9557 | 1.0223 |
| $A_2/A_2'$ | 1.0022 | 1.0000 | 0.9902 | 1.0011 | 1.0152 | 1.0030 | 1.0015 | 1.0048 | 1.0159 | 0.9928 |
| $A_3/A_3'$ | 1.0020 | 0.9959 | 1.0030 | 1.0090 | 1.0001 | 0.9958 | 1.0086 | 1.0154 | 0.9992 | 1.0115 |
| $A_4/A_4'$ | 0.9997 | 0.9982 | 0.9932 | 0.9982 | 1.0030 | 1.0043 | 0.9974 | 0.9965 | 1.0081 | 0.9854 |
| $A_5/A_5'$ | 0.9969 | 0.9662 | 1.0042 | 1.0021 | 1.0297 | 1.0139 | 1.0003 | 1.0208 | 1.0104 | 1.0055 |
| $A_6/A_6'$ | 1.0176 | 1.0405 | 1.0063 | 0.9942 | 0.9634 | 0.9834 | 1.0060 | 0.9752 | 1.0034 | 0.9847 |
| $E^2$ | 647.85 | 664.67 | 569.86 | 655.98 | 669.67 | 615.28 | 570.37 | 579.80 | 626.42 | 663.80 |

Table 6.5 Fitted Results from Ridge Regression with Group-1 Decay Constant Fixed (Saturation Mode)

| seed | -6295779 | -389279151 | 738164549 | -1778841095 | 1234 | -1031501663 | -1141185387 | 774385225 | 397015485 | 534354673 |
|---|---|---|---|---|---|---|---|---|---|---|
| $A_1/A_1'$ | 0.9966 | 1.0057 | 1.0072 | 0.9951 | 1.0019 | 0.9916 | 0.9956 | 1.0033 | 0.9885 | 1.0082 |
| $A_2/A_2'$ | 0.9986 | 1.0069 | 0.9951 | 0.9926 | 1.0113 | 0.9793 | 0.9983 | 1.0104 | 0.9849 | 1.0058 |
| $A_3/A_3'$ | 0.9774 | 1.0930 | 1.0182 | 0.9402 | 1.0239 | 0.8795 | 1.0198 | 0.9854 | 0.8135 | 1.0716 |
| $A_4/A_4'$ | 1.0222 | 0.9833 | 1.0124 | 1.0087 | 0.9766 | 1.0399 | 0.9868 | 0.8659 | 1.0343 | 0.9895 |
| $A_5/A_5'$ | 1.0224 | 0.3781 | 0.9786 | 1.1669 | 0.8473 | 0.9846 | 0.9942 | 1.0984 | 1.3039 | 1.0671 |
| $A_6/A_6'$ | 0.8263 | 2.4928 | 1.0307 | 0.9187 | 1.2912 | 1.2710 | 1.0710 | 1.8720 | 0.9130 | 0.3356 |
| $\lambda_1/\lambda_1'$ | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| $\lambda_2/\lambda_2'$ | 0.9989 | 1.0032 | 1.0002 | 0.9960 | 1.0049 | 0.9895 | 0.9979 | 1.0055 | 0.9922 | 1.0040 |
| $\lambda_3/\lambda_3'$ | 0.9867 | 1.0458 | 0.9961 | 0.9586 | 1.0283 | 0.9127 | 1.0024 | 1.0197 | 0.8934 | 1.0356 |
| $\lambda_4/\lambda_4'$ | 0.9943 | 1.0599 | 1.0228 | 0.9635 | 1.0142 | 0.9375 | 1.0032 | 0.9309 | 0.8979 | 1.0498 |
| $\lambda_5/\lambda_5'$ | 1.1854 | 0.8179 | 1.0619 | 0.9581 | 0.8450 | 0.8693 | 0.9857 | 0.5933 | 0.8775 | 1.3756 |
| $\lambda_6/\lambda_6'$ | 0.6762 | 0.7368 | 1.1333 | 1.8380 | 0.7828 | 0.7405 | 0.9543 | 0.7141 | 1.4615 | 0.5219 |
| $E^2$ | 643.89 | 661.43 | 563.78 | 651.60 | 668.05 | 607.95 | 567.81 | 574.78 | 621.80 | 663.49 |

Table 6.6 Fitted Results from Ridge Regression with Group-1 Parameters Fixed (Pulse Mode)

| seed | -6295779 | -389279151 | 738164549 | -1778841095 | 1234 | -1031501663 | -1141185387 | 774385225 | 397015485 | 534354673 |
|---|---|---|---|---|---|---|---|---|---|---|
| $A_1/A_1'$ | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| $A_2/A_2'$ | 0.9922 | 1.0035 | 0.9804 | 0.9919 | 1.0181 | 0.9603 | 1.0054 | 1.0097 | 0.9811 | 0.9762 |
| $A_3/A_3'$ | 0.9280 | 1.0127 | 1.0329 | 1.0015 | 0.9946 | 0.9216 | 1.0451 | 0.9546 | 0.9227 | 0.8911 |
| $A_4/A_4'$ | 1.0097 | 0.9759 | 0.9960 | 1.0113 | 0.9801 | 1.0276 | 0.9826 | 0.9861 | 1.0439 | 0.9884 |
| $A_5/A_5'$ | 0.9251 | 0.9324 | 1.0000 | 1.0673 | 0.9844 | 0.9672 | 0.9969 | 1.0491 | 1.0780 | 0.9736 |
| $A_6/A_6'$ | 1.4851 | 1.3169 | 0.9701 | 0.7580 | 1.1391 | 1.4417 | 0.9397 | 1.1466 | 0.8545 | 1.7545 |
| $\lambda_1/\lambda_1'$ | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| $\lambda_2/\lambda_2'$ | 1.0005 | 0.9978 | 0.9894 | 0.9969 | 1.0126 | 0.9752 | 1.0022 | 1.0101 | 0.9938 | 0.9876 |
| $\lambda_3/\lambda_3'$ | 0.9562 | 1.0083 | 0.9890 | 0.9818 | 1.0188 | 0.9031 | 1.0282 | 0.9970 | 0.9266 | 0.9156 |
| $\lambda_4/\lambda_4'$ | 0.9601 | 0.9946 | 1.0118 | 1.0007 | 1.0005 | 0.9493 | 1.0166 | 0.9658 | 0.9701 | 0.9194 |
| $\lambda_5/\lambda_5'$ | 0.8492 | 0.8845 | 1.0133 | 1.0621 | 0.9351 | 0.8373 | 1.0327 | 0.9050 | 1.0228 | 0.7213 |
| $\lambda_6/\lambda_6'$ | 0.8477 | 0.9229 | 1.0283 | 1.1471 | 0.9390 | 0.8546 | 1.0223 | 0.9475 | 1.0780 | 0.7883 |
| $E^2$ | 644.61 | 662.00 | 566.77 | 653.61 | 667.07 | 604.43 | 569.36 | 577.16 | 620.25 | 657.77 |

Standard deviations of the fitted results from simulated data sets are calculated to provide information about fitted group parameters distribution. The information is shown in Figures 6.2, 6.3, and 6.4. Figures 6.2 and 6.3 are the standard deviation of the fitted ratios in saturation mode and pulse mode, respectively. Figure 6.4 illustrates the standard deviation when Group-1 parameters are fixed in pulse mode. The data sources in these figures are from the values in Tables 6.1, 6.2 and 6.6, respectively. Group-5 and -6 (shorter half-lived groups) are the most variable groups in saturation mode (Figure 6.2), but in pulse mode (Figure 6.3), the most variable data are from Group-1 and -6. Group-1 has large variation because the Group-1 counts in the later decay period are much lower in pulse mode. Variation for Group-6 is reduced in pulse mode compared to that in saturation mode. The variation from relative yields is larger than decay constants in most cases. This is because the yields reflect a linear relationship in the objective function, while decay constants represent an exponential relationship. The change in exponential part is more significant.

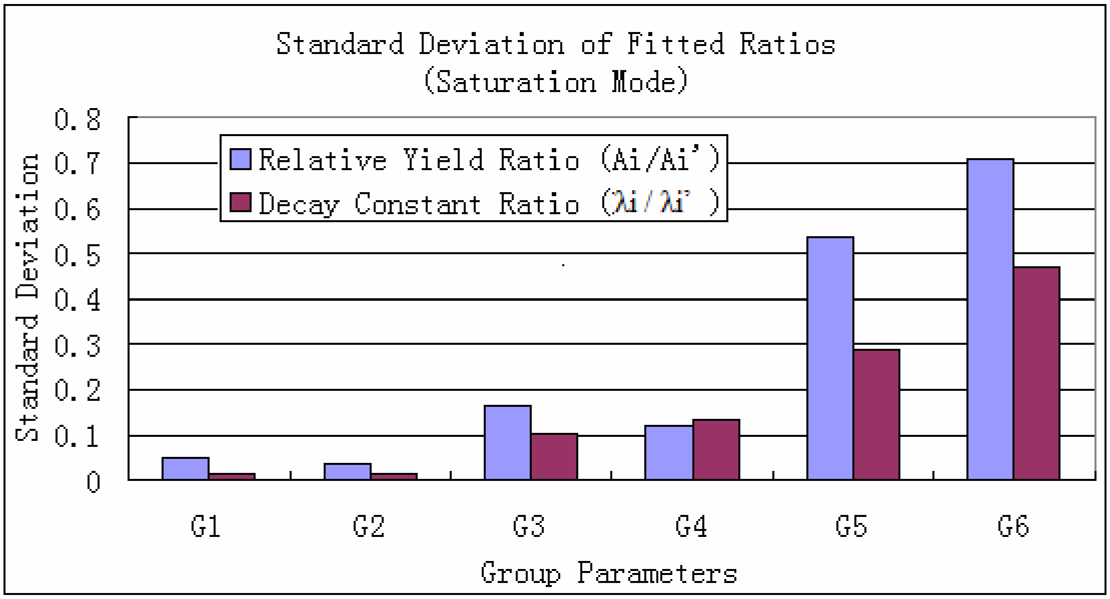


Figure 6.2 Standard Deviations of Fitted Parameter Ratios (Saturation Mode)

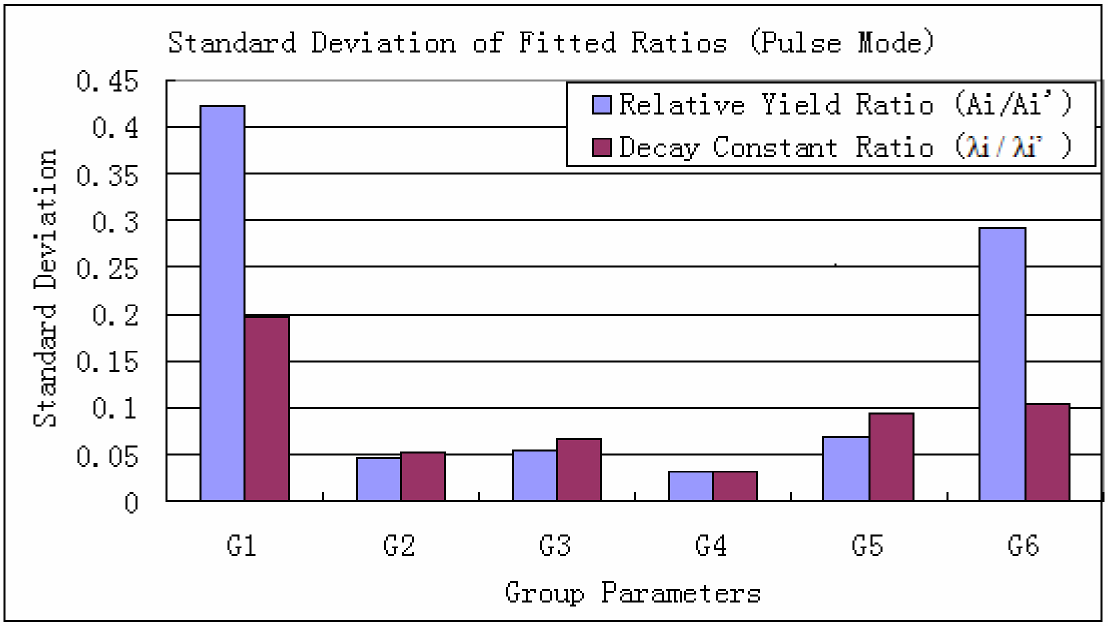


Figure 6.3 Standard Deviations of Fitted Parameter Ratios (Pulse Mode)

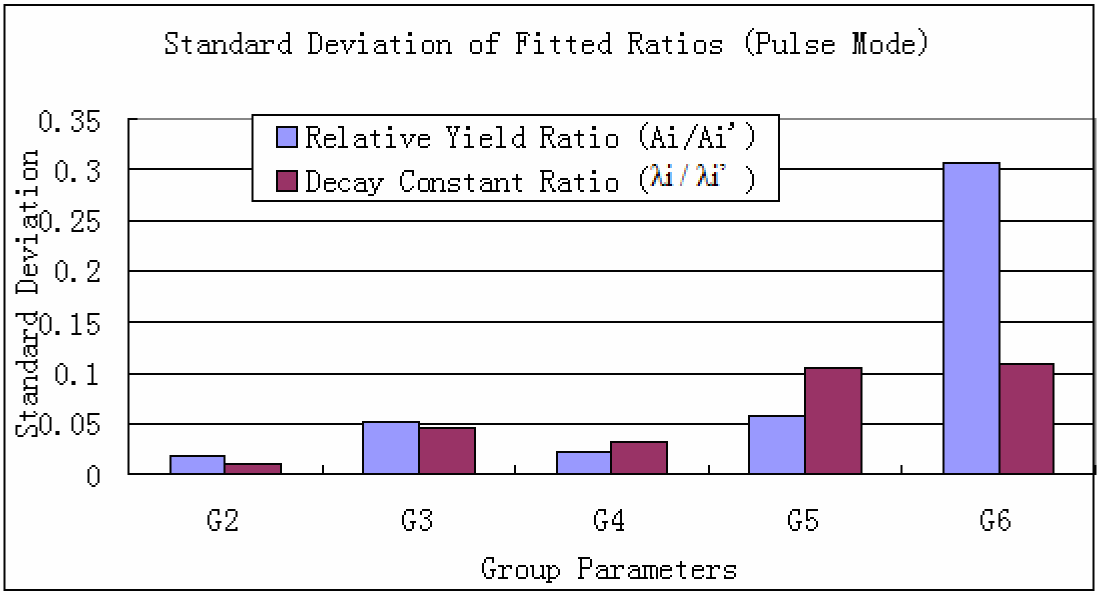


Figure 6.4 Standard Deviations of Fitted Parameter Ratios with Group-1 Fixed (Pulse Mode)

All of the above solutions from different algorithms used the same values (Keepin's values) as initial guessed input, and their solutions are totally different with different $E^2$. Obviously, the extracted values are not global optimum at all. These local optimization values are highly dependent on the initial guessed values. RR always obtains a smaller $E^2$ than other algorithms because it has a broader searching range to find the optimization point. If the initial guessed inputs are changed in different algorithms, the new fitted results may be different. To verify this effect, the extracted values from RR are used as initial guessed values in other algorithms. The results are shown in Table 6.7 (Saturation Mode) and Table 6.8 (Pulse Mode). Only two data sets are listed as examples.

Table 6.7 Fitted Results Using New Initial Values from Ridge Regression (Saturation Mode)

| | | RR | MI-SVD | | LM | | QN | |
|---|---|---|---|---|---|---|---|---|
| | | | Before* | After* | Before | After | Before | After |
| Data set 1# (seed= -6295779) | $A_1/A_1'$ | 0.8863 | 0.9913 | 0.8896 | 0.9687 | 0.8863 | 1.1506 | 0.8846 |
| | $A_2/A_2'$ | 0.9572 | 1.0049 | 0.9558 | 0.9985 | 0.9572 | 1.2426 | 0.9553 |
| | $A_3/A_3'$ | 0.5500 | 0.9875 | 0.5498 | 0.9664 | 0.5500 | 0.7107 | 0.5464 |
| | $A_4/A_4'$ | 0.5972 | 1.0086 | 0.5996 | 1.0307 | 0.5972 | 0.7719 | 0.5934 |
| | $A_5/A_5'$ | 2.4703 | 0.9319 | 2.4609 | 0.8490 | 2.4703 | 3.2238 | 2.4784 |
| | $A_6/A_6'$ | 3.1784 | 1.1487 | 3.1912 | 1.3723 | 3.1784 | 4.1304 | 3.1754 |
| | $\lambda_1/\lambda_1'$ | 0.9679 | 0.9966 | 0.9682 | 0.9907 | 0.9679 | 0.9679 | 0.9679 |
| | $\lambda_2/\lambda_2'$ | 0.9637 | 1.0009 | 0.9638 | 0.9919 | 0.9637 | 0.9637 | 0.9637 |
| | $\lambda_3/\lambda_3'$ | 0.7307 | 1.0040 | 0.7278 | 0.9693 | 0.7307 | 0.7294 | 0.7294 |
| | $\lambda_4/\lambda_4'$ | 0.6110 | 0.9943 | 0.6138 | 0.9851 | 0.6110 | 0.6088 | 0.6088 |
| | $\lambda_5/\lambda_5'$ | 0.3220 | 1.0954 | 0.3209 | 1.0958 | 0.3220 | 0.3213 | 0.3213 |
| | $\lambda_6/\lambda_6'$ | 0.5441 | 0.6925 | 0.5451 | 0.6295 | 0.5441 | 0.5437 | 0.5437 |
| | $E^2$ | 640.52 | 645.14 | 641.69 | 643.18 | 640.52 | 640.52 | 640.52 |
| Data set 2# (seed= -389279151) | $A_1/A_1'$ | 1.0608 | 1.0061 | 1.0713 | 1.0079 | 1.0607 | 1.0352 | 1.0649 |
| | $A_2/A_2'$ | 1.0126 | 0.9985 | 1.0069 | 0.9999 | 1.0126 | 0.9866 | 1.0150 |
| | $A_3/A_3'$ | 1.1436 | 0.9973 | 1.1619 | 1.0113 | 1.1436 | 1.1155 | 1.1476 |
| | $A_4/A_4'$ | 0.9654 | 1.0082 | 0.9486 | 0.9997 | 0.9654 | 0.9414 | 0.9685 |
| | $A_5/A_5'$ | 0.2464 | 0.8598 | 0.3007 | 0.8269 | 0.2464 | 0.2395 | 0.2464 |
| | $A_6/A_6'$ | 2.7052 | 1.4731 | 2.6562 | 1.5301 | 2.7049 | 2.6232 | 2.6987 |
| | $\lambda_1/\lambda_1'$ | 1.0160 | 0.9997 | 1.0193 | 1.0019 | 1.0161 | 1.0166 | 1.0166 |
| | $\lambda_2/\lambda_2'$ | 1.0160 | 1.0005 | 1.0153 | 1.0013 | 1.0160 | 1.0163 | 1.0163 |
| | $\lambda_3/\lambda_3'$ | 1.0874 | 0.9888 | 1.0805 | 1.0055 | 1.0874 | 1.0882 | 1.0882 |
| | $\lambda_4/\lambda_4'$ | 1.0929 | 1.0069 | 1.1005 | 1.0104 | 1.0929 | 1.0942 | 1.0942 |
| | $\lambda_5/\lambda_5'$ | 0.7373 | 0.9948 | 0.6674 | 0.9886 | 0.7372 | 0.7527 | 0.7527 |
| | $\lambda_6/\lambda_6'$ | 0.7221 | 1.0574 | 0.7403 | 1.0047 | 0.7221 | 0.7237 | 0.7237 |
| | $E^2$ | 660.08 | 667.66 | 665.74 | 662.67 | 660.11 | 660.10 | 660.10 |

*Note:

Before: Using Keepin's data as initial guessed values;
After: Using RR fitted data as initial guessed values.

Table 6.8 Fitted Results Using New Initial Values from Ridge Regression (Pulse Mode)

| | | RR | LM | | QN | |
|---|---|---|---|---|---|---|
| | | | Before | After | Before | After |
| Data set 1# (seed= -6295779) | $A_1/A_1'$ | 0.6782 | 0.8149 | 0.6286 | 0.5995 | 0.6143 |
| | $A_2/A_2'$ | 0.9854 | 1.0013 | 0.9925 | 0.9716 | 0.9957 |
| | $A_3/A_3'$ | 0.8977 | 0.9629 | 0.8983 | 0.8791 | 0.9009 |
| | $A_4/A_4'$ | 1.0334 | 1.0205 | 1.0322 | 1.0132 | 1.0383 |
| | $A_5/A_5'$ | 0.9604 | 0.9450 | 0.9589 | 0.9429 | 0.9663 |
| | $A_6/A_6'$ | 1.5876 | 1.2284 | 1.6014 | 1.5616 | 1.6004 |
| | $\lambda_1/\lambda_1'$ | 0.8936 | 0.9285 | 0.8278 | 0.8139 | 0.8139 |
| | $\lambda_2/\lambda_2'$ | 0.9419 | 0.9730 | 0.9366 | 0.9331 | 0.9331 |
| | $\lambda_3/\lambda_3'$ | 0.8699 | 0.9492 | 0.8683 | 0.8635 | 0.8635 |
| | $\lambda_4/\lambda_4'$ | 0.9267 | 0.9783 | 0.9257 | 0.9245 | 0.9245 |
| | $\lambda_5/\lambda_5'$ | 0.7941 | 0.9475 | 0.7893 | 0.7904 | 0.7904 |
| | $\lambda_6/\lambda_6'$ | 0.8284 | 0.9068 | 0.8239 | 0.8270 | 0.8270 |
| | $E^2$ | 642.07 | 643.73 | 642.11 | 642.07 | 642.07 |
| Data set 2# (seed= -389279151) | $A_1/A_1'$ | 1.3175 | 1.3650 | 1.3538 | 1.2996 | 1.3019 |
| | $A_2/A_2'$ | 0.9947 | 0.9977 | 0.9918 | 0.9501 | 0.9517 |
| | $A_3/A_3'$ | 1.0602 | 1.0692 | 1.0615 | 1.0178 | 1.0196 |
| | $A_4/A_4'$ | 0.9456 | 0.9515 | 0.9463 | 0.9062 | 0.9078 |
| | $A_5/A_5'$ | 0.9027 | 0.9086 | 0.9036 | 0.8654 | 0.8669 |
| | $A_6/A_6'$ | 1.2573 | 1.2647 | 1.2602 | 1.2052 | 1.2073 |
| | $\lambda_1/\lambda_1'$ | 1.0992 | 1.1168 | 1.1160 | 1.1168 | 1.1168 |
| | $\lambda_2/\lambda_2'$ | 1.0499 | 1.0540 | 1.0535 | 1.0539 | 1.0539 |
| | $\lambda_3/\lambda_3'$ | 1.0845 | 1.0874 | 1.0862 | 1.0870 | 1.0870 |
| | $\lambda_4/\lambda_4'$ | 1.0277 | 1.0288 | 1.0280 | 1.0285 | 1.0285 |
| | $\lambda_5/\lambda_5'$ | 0.9252 | 0.9265 | 0.9249 | 0.9261 | 0.9261 |
| | $\lambda_6/\lambda_6'$ | 0.9372 | 0.9377 | 0.9369 | 0.9375 | 0.9375 |
| | $E^2$ | 660.18 | 660.28 | 660.28 | 660.28 | 660.28 |

*Note:

Before: Using Keepin’s data as initial guessed values;
After: Using RR values as initial guessed values.
MI-SVD is not used in pulse mode.

As shown in the above tables, if the extracted values from RR are used as new initial guessed values in other algorithms, they can provide almost the same results as that from RR, except for MI-SVD. MI-SVD can secure a smaller $E^2$ compared to its previous result, but it is still larger than those from other algorithms. LM and QN can attain almost the same optimization point if initial guessed values are chosen from RR fitted values. This evidence shows that these algorithms are quite dependent on the initial guessed values for this problem, especially the MI-SVD and LM methods.

Since no numerical methods so far can guarantee finding a global minimum solution for this problem, the initial guessed input values would affect the search range to find a local minimum point. A stable solution would not depend on the initial input values, but for an unstable solution, its value would be sensitive to the guessed input value. The fitted values for Group-6 abundance are highly variable in both modes, but they all fit the simulated experimental data very well. Here, a small change in the initial guessed $A_6$ is applied to check its sensitivity on the fitted results. RR and LM algorithms are used for comparison. Six randomly selected data sets in saturation mode are used as examples.

In the LM algorithm, the initial guessed value for Group-6 relative yield $A_6$ is increased by only 5% for different simulated measurement data. The relative changes for the new fitted results are shown in Table 6.9. In the RR method, a 10% increase is applied, instead of 5%, due to the very small changes in the results from a 5% increase in this method. The relative changes in the results are shown in Table 6.10.

$\Delta A_1/A_1'$ and $\Delta\lambda_1/\lambda_1'$ are the relative changes of the ratios for relative yield and decay constant of group i. From the tables, the fitted objective values $E^2$ are almost the same ($\Delta E^2/ E^2 \approx 0$), even after a 10% change in the initial guess input using the RR method. While from LM method, with even a 5% input change, the relative changes in the new fitted values are much larger than those for RR with 10% guessed input change. This means RR is less sensitive than LM to initial guessed input value $A_6$ and can find relatively more stable $E^2$ than LM. The fitted results for Group-1, -2, -3, and -4 are not changed or just changed a small amount. For Group-5 and -6, the changes are larger. The relative changes in the results from the Matrix Inverse method are very large and the results are not listed here.

As mentioned in Section 5, the introduction of the regularization parameter h in the solution brings bias into the solution. The traditional statistical error in the model using the least squares method is simply the estimate of variance. In the Ridge Regression method, the mean squared error is the sum of the errors from variance and squared bias (see Eq.(5-17)). A properly selected regularization parameter h can achieve a smaller error than the traditional least squares method.

Table 6.9 Fitted Results from Levenberg-Marquardt by Increasing Guessed Value $A_6$ with 5% (Saturation Mode)

| seed | -6295779 | -389279151 | 738164549 | -1778841095 | 1234 | -1031501663 |
|---|---|---|---|---|---|---|
| $\Delta A_1/A_1'$ | 0.282% | 0.086% | 0.025% | 0.890% | 0.025% | 1.510% |
| $\Delta A_2/A_2'$ | 0.031% | 0.007% | 0.050% | 0.162% | 0.013% | 0.115% |
| $\Delta A_3/A_3'$ | 0.678% | 0.157% | 0.029% | 5.488% | 0.010% | 7.240% |
| $\Delta A_4/A_4'$ | 0.168% | 0.089% | 0.068% | 2.014% | 0.030% | 9.503% |
| $\Delta A_5/A_5'$ | 5.429% | 2.710% | 0.140% | 7.957% | 0.639% | 4.455% |
| $\Delta A_6/A_6'$ | 11.897% | 3.615% | 0.947% | 5.404% | 2.034% | 47.198% |
| $\Delta\lambda_1/\lambda_1'$ | 0.090% | 0.027% | 0.010% | 0.463% | 0.010% | 0.370% |
| $\Delta\lambda_2/\lambda_2'$ | 0.092% | 0.024% | 0.010% | 0.581% | 0.003% | 0.200% |
| $\Delta\lambda_3/\lambda_3'$ | 0.493% | 0.111% | 0.059% | 4.012% | 0.020% | 2.456% |
| $\Delta\lambda_4/\lambda_4'$ | 0.439% | 0.090% | 0.062% | 3.763% | 0.016% | 7.850% |
| $\Delta\lambda_5/\lambda_5'$ | 2.976% | 0.970% | 0.343% | 5.589% | 0.442% | 95.753% |
| $\Delta\lambda_6/\lambda_6'$ | 0.168% | 3.602% | 0.039% | 5.509% | 0.463% | 18.172% |
| $\Delta E^2/E^2$ | 0.003% | 0.017% | 0.001% | 0.060% | 0.008% | 0.079% |

Table 6.10 Fitted Results from Ridge Regression by Increasing Guessed Value $A_6$ with 10% (Saturation Mode)

| seed | -6295779 | -389279151 | 738164549 | -1778841095 | 1234 | -1031501663 |
|---|---|---|---|---|---|---|
| $\Delta A_1/A_1'$ | 0.169% | 0.023% | 0.000% | 0.000% | 0.041% | 0.196% |
| $\Delta A_2/A_2'$ | 0.141% | 0.010% | 0.001% | 0.000% | 0.020% | 0.087% |
| $\Delta A_3/A_3'$ | 2.589% | 0.150% | 0.010% | 0.001% | 0.262% | 1.403% |
| $\Delta A_4/A_4'$ | 1.954% | 0.064% | 0.010% | 0.000% | 0.152% | 0.714% |
| $\Delta A_5/A_5'$ | 2.651% | 3.135% | 0.127% | 0.009% | 0.070% | 8.773% |
| $\Delta A_6/A_6'$ | 0.411% | 1.072% | 0.396% | 0.009% | 1.967% | 16.433% |
| $\Delta\lambda_1/\lambda_1'$ | 0.043% | 0.010% | 0.000% | 0.000% | 0.011% | 0.049% |
| $\Delta\lambda_2/\lambda_2'$ | 0.072% | 0.010% | 0.000% | 0.000% | 0.014% | 0.079% |
| $\Delta\lambda_3/\lambda_3'$ | 1.141% | 0.064% | 0.003% | 0.000% | 0.129% | 0.688% |
| $\Delta\lambda_4/\lambda_4'$ | 1.878% | 0.145% | 0.010% | 0.001% | 0.238% | 1.181% |
| $\Delta\lambda_5/\lambda_5'$ | 1.081% | 0.187% | 0.126% | 0.004% | 1.227% | 11.981% |
| $\Delta\lambda_6/\lambda_6'$ | 0.274% | 0.753% | 0.224% | 0.070% | 0.621% | 4.093% |
| $\Delta E^2/E^2$ | 0.002% | 0.005% | 0.000% | 0.000% | 0.001% | 0.013% |

The differences of root of mean squared error (RMSE) from these two methods are shown in Table 6.11 (saturation mode) and Table 6.12 (pulse mode). One random number (seed=1234) is used for data generation, and the initial count rates are the same in both modes (e.g., 4.0E+5 cps) before randomization. The calculated relative error (RMSE-RR/X*) in saturation mode has larger uncertainty for short-lived groups (G-5 and G-6). For short-lived groups, the relative errors from pulse mode are smaller than those from saturation mode using the same initial count rate, but for Group-1, it increases. Compared to the least squares method, RMSE from RR is reduced with the help of regularization parameter h. This role is more significant for G-5 & G-6 in both modes, due to the smaller singular values from the matrices in the modes. Researchers previously used the inverse of the matrix from the normal equation to obtain the statistical error. This is not correct and it is only valid for a linear model without bias. For those who used LM method, or some other methods related to a regularization number, they did not consider the bias in the error estimation at all.

As shown in the tables above, there are several reasons that RR is better than other algorithms (MI-SVD, LM, and QN). RR has the following advantages:

1) It achieves better fit illustrated by its smaller $E^2$ values for the same data set.
2) It converges if parameter h is properly selected.
3) It is not as sensitive as other algorithms to the initial guessed values.

Table 6.11 Root of Mean Squared Error (Saturation Mode)

| | X* | RMSE-RR<br>(Ridge Regression) | RMSE-LS<br>(Least Squares) | RMSE-RR/X* |
|---|---|---|---|---|
| $A_1$ | 3.22E-02 | 1.77E-03 | 6.28E-03 | 5.49% |
| $A_2$ | 2.21E-01 | 2.78E-03 | 3.07E-03 | 1.26% |
| $A_3$ | 1.96E-01 | 2.44E-02 | 3.54E-02 | 12.49% |
| $A_4$ | 3.88E-01 | 2.11E-02 | 4.06E-02 | 5.44% |
| $A_5$ | 9.96E-02 | 3.57E-02 | 8.63E-02 | 35.88% |
| $A_6$ | 5.62E-02 | 2.40E-02 | 1.32E-01 | 42.64% |
| $\lambda_1$ | 1.23E-02 | 1.97E-04 | 9.57E-04 | 1.60% |
| $\lambda_2$ | 3.05E-02 | 4.31E-04 | 1.01E-03 | 1.42% |
| $\lambda_3$ | 1.12E-01 | 8.27E-03 | 1.25E-02 | 7.39% |
| $\lambda_4$ | 3.00E-01 | 2.63E-02 | 4.48E-02 | 8.76% |
| $\lambda_5$ | 9.42E-01 | 2.86E-01 | 1.03E+00 | 30.38% |
| $\lambda_6$ | 2.35E+00 | 7.33E-01 | 2.71E+00 | 31.19% |

Table 6.12 Root of Mean Squared Error (Pulse Mode)

| | X* | RMSE-RR<br>(Ridge Regression) | RMSE-LS<br>(Least Squares) | RMSE-RR/X* |
|---|---|---|---|---|
| $A_1$ | 2.67E-02 | 6.75E-03 | 1.52E-02 | 25.30% |
| $A_2$ | 2.28E-01 | 5.12E-03 | 1.14E-02 | 2.24% |
| $A_3$ | 1.96E-01 | 1.64E-02 | 1.67E-02 | 8.34% |
| $A_4$ | 3.90E-01 | 1.39E-02 | 1.60E-02 | 3.57% |
| $A_5$ | 1.14E-01 | 8.52E-03 | 8.65E-03 | 7.48% |
| $A_6$ | 4.56E-02 | 1.35E-02 | 1.37E-02 | 29.51% |
| $\lambda_1$ | 1.03E-02 | 2.12E-03 | 6.40E-03 | 20.64% |
| $\lambda_2$ | 3.02E-02 | 1.16E-03 | 2.29E-03 | 3.84% |
| $\lambda_3$ | 1.12E-01 | 8.71E-03 | 1.07E-02 | 7.78% |
| $\lambda_4$ | 3.02E-01 | 1.48E-02 | 1.60E-02 | 4.91% |
| $\lambda_5$ | 1.09E+00 | 1.48E-01 | 1.52E-01 | 13.64% |
| $\lambda_6$ | 2.88E+00 | 3.38E-01 | 3.42E-01 | 11.76% |

Note: X*: Fitted parameter results from Ridge Regression method.

4) RMSE from RR is much smaller than from the ordinary least squares method, especially for short-lived groups.

The calculated results in the above tables also reveal some other facts:

1) All algorithms mentioned above cannot find global optimum. The fitted solutions are highly variable.

2) In saturation mode, fitted results from Group-6 have very large variation. In pulse mode, the larger variations are from Group-1 and Group-6. Group-6 variation is still very large but it is reduced compared to that in saturation mode. Group-1 variation increases because the count from Group-1 is much smaller than that from saturation mode, and this increase uncertainty.

3) Fixing some parameters does not improve the optimization effect based on the simulated results. Let all parameters free, the optimization results are the best.

## 6.2 Data Comparison

Since all the extracted parameters fit the simulated data sets very well even if they have very large discrepancies, it is not possible to judge the goodness-of-fit between investigators just by the fitted parameters themselves. One should compare the difference between data sets to see if they are significantly different or not. Figure 6.5 plots the ratios of four fitted data sets from RR, as well as the original Poisson data, to Keepin’s input values for saturation mode. Figure 6.6 is for pulse mode with Group-1 parameters fixed. These figures show that the fitted data from RR fit Keepin's input data very well. In saturation mode, after about 100 seconds, there is a small departure from the ratio of 1.0 because only Group-1 is dominant in this period and the fitted parameters are not exactly the same as Keepin's values, due to the limited counts used for fitting.

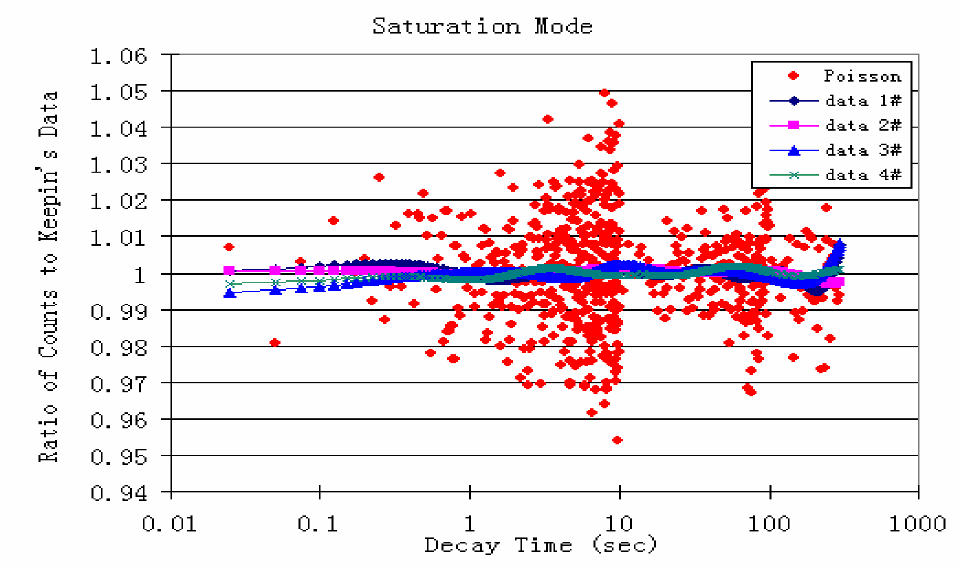


Figure 6.5 The Ratios of Different Data Sets to Keepin's Values (Saturation Mode)

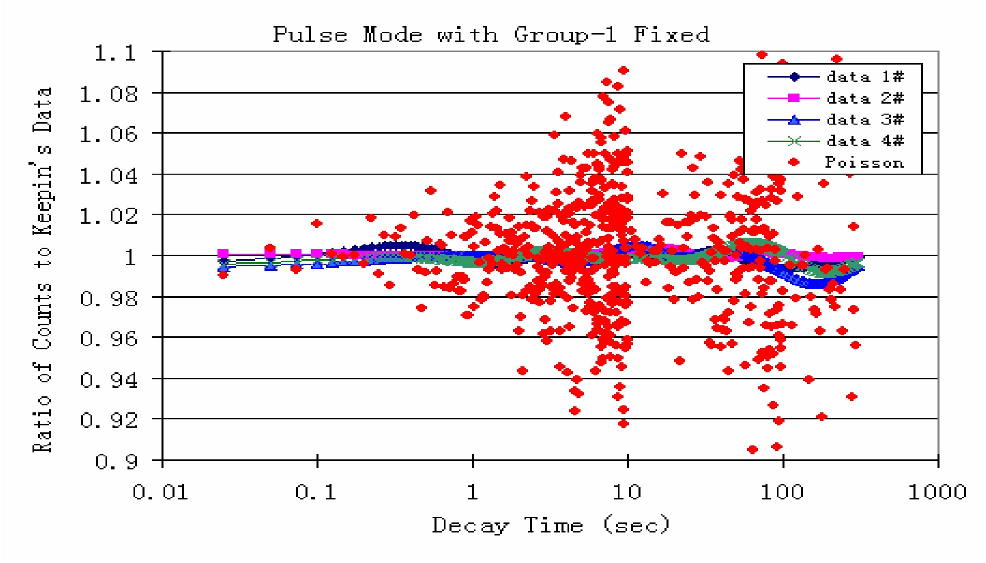


Figure 6.6 The Ratios of Different Data Sets to Keepin's Values (Pulse Mode with Group-1 Fixed)

In this section, the data sets are analyzed from the following components:

1) The comparison between the fitted data and simulated counts. The purpose is to check if the fitted results are significantly different from the Poisson distributed simulated count. The comparison between Poisson data and Keepin's data will not be analyzed here because it has already been done in the Data Generation part of Section 4.

2) The comparison between the fitted data and Keepin's data. This is to check if the fitted data are still the same as Keepin's original data.

3) The comparison between the newly fitted results with $A_6$ increased by 10% and Keepin's original data with zero flight time. $A_6$ is chosen because it is the most variable value among the fitted parameters and it

still fit well by the data set. The purpose here is to discover if a small change in $A_6$ can be differentiated from Keepin's original data.

4) The comparison between new fitted data with $A_6$ increased by 10% and Keepin's original data under different sample flight time and initial count rate. It is very helpful in a real experiment to see how much flight time is acceptable for a given experiment.

To obtain more detailed information at different decay times, the total counting time (300 sec) is divided into several time intervals according to the count contribution from different groups. Both saturation and pulse modes are analyzed. A Chi-square test is used to compare the difference of sample data from the expected data. If $\chi^2 > \chi^2_{0(\alpha,n-1)}$, that means the sample data are significantly different from the expected data under given confidence level 100(1-α)% with n-1 degrees of freedom. $\chi^2_{0(\alpha,n-1)}$ is the reference value.

The comparisons between RR data sets and Poisson data, and RR data and Keepin's data are listed in Table 6.13 (saturation mode) and Table 6.14 (pulse mode) with significance level α=0.05. Data sets 1#, 2#, 3# and 4# are generated from seeds –6295779, -389279151, -1031501663, and 4250 respectively. They were chosen because their fitted values $A_6$ have larger discrepancies against Keepin's values.

All $\chi^2$ values in Table 6.13 and Table 6.14 are smaller than the reference values $\chi^2_{0(\alpha,n-1)}$. This means that the fitted data sets from RR are not significantly different from the corresponding Poisson distributed data sets, nor are they from

Keepin's data. As $\chi^2$ values are very small in the comparisons of RR data sets and Keepin's data, it shows that the data sets from RR fit Keepin's data very well.

Table 6.13 Difference Comparison of RR Data to Poisson and Keepin's Data (Saturation Mode) (α=0.05)

| Time (s) | RR & Poisson data | | | | RR & Keepin | | | | n | $\chi_0^2$(α,n-1) |
|---|---|---|---|---|---|---|---|---|---|---|
| | data 1# | data 2# | data 3# | data 4# | data 1# | data 2# | data 3# | data 4# | | |
| 0-1 | 53.43 | 40.20 | 30.92 | 31.81 | 1.04 | 0.04 | 1.29 | 0.90 | 40 | 55.76 |
| 1-2 | 35.07 | 30.81 | 39.27 | 48.40 | 0.68 | 0.09 | 0.02 | 0.47 | 40 | 55.76 |
| 2-4 | 81.63 | 79.24 | 72.84 | 91.95 | 0.15 | 0.06 | 0.33 | 0.47 | 80 | 101.88 |
| 4-6 | 74.23 | 75.79 | 90.85 | 92.87 | 0.05 | 0.04 | 0.50 | 0.15 | 80 | 101.88 |
| 6-8 | 81.59 | 78.46 | 62.64 | 73.64 | 0.07 | 0.23 | 0.16 | 0.04 | 80 | 101.88 |
| 8-10 | 98.43 | 72.67 | 83.91 | 64.44 | 0.10 | 0.37 | 0.92 | 0.09 | 80 | 101.88 |
| 10-40 | 68.77 | 61.68 | 58.45 | 58.38 | 0.68 | 1.17 | 2.04 | 0.43 | 60 | 79.08 |
| 40-80 | 68.96 | 66.00 | 60.89 | 51.90 | 0.71 | 0.04 | 0.24 | 2.31 | 80 | 101.88 |
| 80-300 | 77.87 | 63.07 | 74.63 | 67.60 | 2.36 | 0.76 | 2.63 | 0.28 | 80 | 101.88 |

Table 6.14 Difference Comparison of RR Data to Poisson and Keepin's Data (Pulse Mode) (α=0.05)

| Time (s) | RR & Poisson data | | | | RR & Keepin | | | | n | $\chi_0^2$(α,n-1) |
|---|---|---|---|---|---|---|---|---|---|---|
| | data 1# | data 2# | data 3# | data 4# | data 1# | data 2# | data 3# | data 4# | | |
| 0-1 | 52.93 | 40.42 | 31.00 | 31.88 | 2.43 | 0.04 | 1.32 | 1.11 | 40 | 55.76 |
| 1-2 | 35.32 | 30.96 | 39.36 | 48.03 | 1.65 | 0.11 | 0.01 | 0.76 | 40 | 55.76 |
| 2-4 | 81.62 | 78.99 | 73.00 | 92.47 | 0.11 | 0.13 | 0.48 | 0.40 | 80 | 101.88 |
| 4-6 | 74.48 | 77.18 | 91.34 | 92.25 | 0.02 | 0.02 | 0.75 | 0.05 | 80 | 101.88 |
| 6-8 | 81.59 | 77.90 | 63.09 | 74.00 | 0.01 | 0.07 | 0.10 | 0.12 | 80 | 101.88 |
| 8-10 | 98.43 | 73.89 | 82.74 | 64.86 | 0.05 | 0.14 | 0.69 | 0.12 | 80 | 101.88 |
| 10-40 | 68.51 | 61.76 | 58.18 | 58.51 | 0.35 | 1.23 | 1.64 | 0.51 | 60 | 79.08 |
| 40-80 | 70.43 | 66.83 | 62.51 | 52.07 | 0.44 | 0.04 | 0.22 | 1.49 | 80 | 101.88 |
| 80-300 | 81.55 | 63.70 | 75.31 | 67.70 | 0.89 | 0.02 | 2.91 | 0.61 | 80 | 101.88 |

Although the extracted Group-6 values are very variable, the final fitted data sets still fit Keepin's data set very well. However, if a small artificial change of count is introduced, for example, if $A_6$ is increased by 10% in new Poisson data generation, the new data can be differentiated from Keepin's original data (see Table 6.15). A Chi-square test is used to analyze the data distribution of the new Poisson data set against Keepin's original data during different decay periods at a 95% confidence level. The initial count rate in Keepin's data is variable to fit the new Poisson data set using SOLVER tool in Microsoft Excel. The initial count rate to generate Poisson data set was set at 4.0E+5 cps before randomization.

Table 6.15 Difference Comparison between Poisson Data and Keepin's Data ($A_6$ Increased by 10%) ($\alpha$=0.05)

| Time (s) | Saturation Mode | | | | Pulse Mode | | | | n | $\chi_0^2(\alpha,n-1)$ |
|---|---|---|---|---|---|---|---|---|---|---|
| | data 1# | data 2# | data 3# | data 4# | data 1# | data 2# | data 3# | data 4# | | |
| 0-1 | 34.62 | 50.37 | 46.91 | 34.26 | 81.94 | 66.50 | 71.98 | 64.50 | 40 | 55.76 |
| 1-2 | 24.58 | 40.53 | 43.12 | 39.83 | 29.71 | 23.41 | 35.98 | 65.78 | 40 | 55.76 |
| 2-4 | 96.58 | 104.54 | 105.55 | 106.06 | 76.72 | 107.31 | 66.73 | 77.82 | 80 | 101.88 |
| 4-6 | 104.70 | 52.97 | 106.82 | 81.57 | 59.27 | 67.34 | 107.77 | 65.08 | 80 | 101.88 |
| 6-8 | 78.18 | 79.43 | 73.89 | 49.04 | 80.04 | 110.49 | 74.71 | 94.08 | 80 | 101.88 |
| 8-10 | 69.84 | 69.62 | 88.09 | 74.44 | 114.30 | 98.56 | 56.04 | 92.62 | 80 | 101.88 |
| 10-40 | 66.20 | 34.59 | 79.60 | 49.49 | 46.76 | 84.74 | 63.81 | 67.45 | 60 | 79.08 |
| 40-80 | 106.83 | 87.00 | 104.79 | 71.92 | 64.35 | 59.17 | 88.44 | 94.66 | 80 | 101.88 |
| 80-300 | 89.03 | 70.22 | 76.72 | 68.37 | 93.32 | 80.29 | 59.52 | 96.02 | 80 | 101.88 |

The calculated $\chi^2$ values in Table 6.15 shows that if $A_6$ is 10% larger than Keepin's original value, the newly simulated data set can be differentiated from Keepin's data. This small artificial change causes significant difference from Keepin's theoretical data in both modes, but it is more obvious in pulse mode. This is expected because it was $A_6$ that was increased and pulse mode favors the signal from the short-lived groups.

All the simulated data sets mentioned above have zero sample flight time, and the same initial counts rate before randomization. In real experiments, the sample flight times are not zero, and initial count rates are higher or lower depending on the setup. The purpose here is to find how much initial count rate and flight time affect our ability to identify differences between the experimental data and Keepin's expected data. To create this difference, a small artificial change is added to $A_6$ under different flight time and initial count rate to check if it can be differentiated or not. Since the short-lived groups are more sensitive to the sample flight time and initial count rate, a small change on $A_6$ (increase by 10%) is chosen to create the difference. A Chi-square test is used to judge the difference between the newly generated data set and Keepin's expected data. One may increase the flight time or decrease the initial count rate until this difference is no longer significant. For a given 95% confidence level, the significant difference regions in both irradiation modes under different flight times and initial count rates are plotted in Figure 6.7. To derive Keepin's data for comparison, the initial count rate is adjusted so that both Keepin's data and newly fitted RR data have a satisfactory fit after time t=5.0 sec. This 5.0 sec is chosen because the counts from Group-6 have decayed away at this time and the 10% change on $A_6$ will not affect the total counts in the later period.

If the initial count rate is very low, the difference can only be identified for a very short period of time. With the increase of flight time, this difference will disappear gradually because of the decay of the short-lived groups. When flight time is too long, or the initial count rate is too low, the two data sets (new Poisson data and Keepin's data) are not significantly different under a 95% confidence level. The influence of this 10% change is more sensitive in pulse mode than in saturation mode because the fraction of count from Group-6 in pulse mode is much higher than in saturation mode at the beginning decay time and its influence lasts much longer.

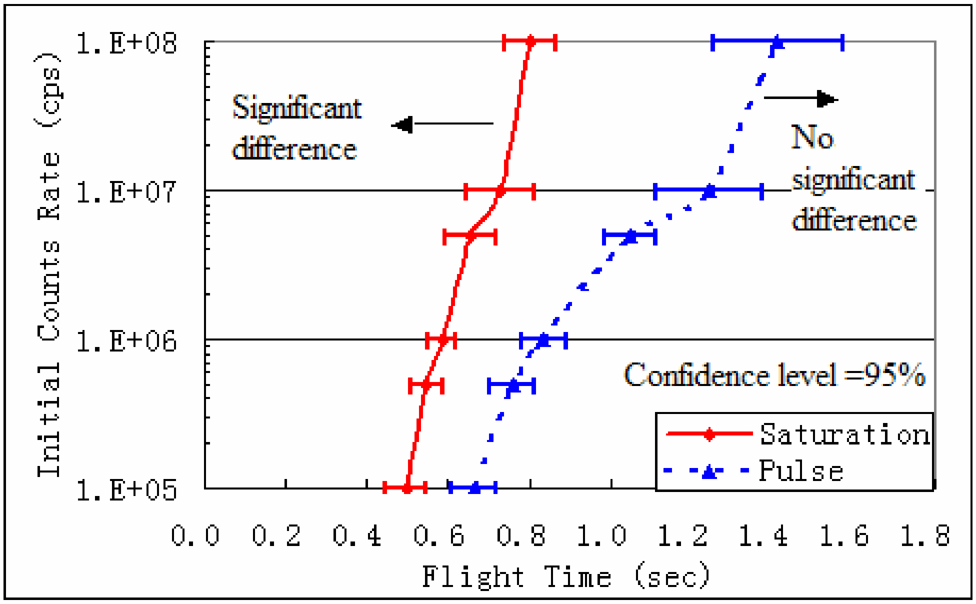


Figure 6.7 Differences between Fitted Data from RR (with $A_6$ Increased by 10%) and Keepin's Data

## 6.3 Experiment Result

All of the calculated results in the above sections are based on simulated data. To apply the above methods in a real experiment, two experimental data sets are used here for verification. The experiment was conducted at the TRIGA reactor at Texas A&M University for both saturation and pulse irradiation modes. Detailed information can be found in Heinrich [62]. The saturation mode has a flight time 0.443 sec, and the pulse mode has a flight time 0.45 sec. The experimental data sets are listed in Appendix C. The experimental data are compared to Keepin's expected values to check differences.

Because of dead time in the detectors, the measured counts must be corrected. If the measured count rate in the detector for the i$^{th}$ observation is *C(i)*, the dead time for the detector is τ sec, and the corrected count rate *C'(i)* is derived by:

$$C'(i) = \frac{C(i)}{1 - C(i) * \tau} \qquad (6\text{-}1)$$

To compare the measured counts to Keepin's theoretical counts, three unknown parameters need to be fitted: dead time τ, initial count rate $N_0$, and an optimized sample flight time $t_0$. Although the sample flight time was measured in the experiment, there was still a small difference between the sample's arrival time and the detector's initial responding time. If these three unknown parameters remain variable, they can be fitted in Microsoft Excel by the SOLVER tool. The objective optimization value is the sum of the weighted squared difference between the measured data and Keepin's data. After optimization, the initial count rate in saturation mode is 4.834E+4 cps, and in pulse mode, it is 2.760E+5 cps. The dead time in saturation mode is 1.01E-6 sec, and in pulse mode it is 6.01E-6 sec. The fitted sample flight time is 0.45sec in both modes. The fitted decay parameters using different algorithms are listed in Table 6.16 for comparison. From the fitted $E^2$ values, the RR algorithm is the best among these methods because it can achieve the smallest $E^2$ in saturation mode. The results in the table below further support the conclusion that the RR algorithm can achieve a better optimization result because its $E^2$ value is smaller.

Table 6.16 The Fitted Results from Experimental Data

| | Saturation | | | | Pulse | | |
|---|---|---|---|---|---|---|---|
| | MI-SVD | LM | QN | RR | LM | QN | RR |
| $A_1/A_1'$ | 0.896 | 1.017 | 1.163 | 1.197 | 1.421 | 1.410 | 1.807 |
| $A_2/A_2'$ | 1.031 | 1.024 | 1.025 | 1.057 | 1.108 | 1.099 | 1.035 |
| $A_3/A_3'$ | 0.947 | 0.584 | 1.246 | 1.274 | 1.290 | 1.280 | 1.283 |
| $A_4/A_4'$ | 1.012 | 1.064 | 0.904 | 0.935 | 0.889 | 0.882 | 0.857 |
| $A_5/A_5'$ | 1.082 | 1.227 | 0.923 | 0.968 | 0.619 | 0.614 | 0.597 |
| $A_6/A_6'$ | 2.290 | 4.562 | 0.859 | 0.856 | 1.030 | 1.021 | 0.995 |
| $\lambda_1/\lambda_1'$ | 0.978 | 0.998 | 1.070 | 1.069 | 1.116 | 1.116 | 1.286 |
| $\lambda_2/\lambda_2'$ | 1.019 | 0.992 | 1.091 | 1.091 | 1.118 | 1.118 | 1.163 |
| $\lambda_3/\lambda_3'$ | 1.016 | 0.657 | 1.257 | 1.252 | 1.307 | 1.307 | 1.338 |
| $\lambda_4/\lambda_4'$ | 0.955 | 0.715 | 1.150 | 1.143 | 1.212 | 1.213 | 1.227 |
| $\lambda_5/\lambda_5'$ | 0.759 | 1.260 | 1.410 | 1.393 | 1.123 | 1.123 | 1.143 |
| $\lambda_6/\lambda_6'$ | 0.979 | 0.160 | 0.534 | 0.538 | 0.731 | 0.732 | 0.727 |
| $E^2$ | 616.64 | 612.08 | 607.95 | 607.94 | 667.99 | 667.98 | 667.60 |

The comparison between the experimental data and other data sets are plotted in Figure 6.8 and Figure 6.9. The fitted data from RR, the Keepin's theoretical data, and ENDF/B-VII.0 data are plotted using the same initial value (same value of P in Eq.(4-1)).

From Figure 6.8 and 6.9, the fitted curve from RR has a good fit with the experimental data in both modes, which can also be proven by the Chi-square test values in Tables 6.17 and 6.18. Keepin's theoretical data are very close to these values. As mentioned in Table 2.7 in Section 2, different data libraries have totally different values. Here, ENDF/B-VII.0 is taken as an example because it is the latest version for DN parameters in the U.S. From the plots below, the curve of ENDF/B-VII.0 is very different from both the Keepin's and the experimental data curves. The experimental data are closer to the Keepin's expected data.

To check if the experimental data are different from Keepin's values or not, their count distributions in different periods are compared. The variance comparison is done by a Chi-square test at a 95% confidence level. Another comparison is the total counts in a given time period. If the counts are significantly different from Keepin's data during the given time period, the absolute value of the difference should be larger than 1.96σ under 95% confidence level. Here, σ is the random variation in a count. Because of the low detection efficiency (including intrinsic efficiency and geometry efficiency), random variation can be taken as the square root of the measurement count even for counts from long counting time compared to the group half-lives [64]. Table 6.17 lists the comparison in saturation mode. According to the Chi-square test, in most part of time intervals, the experimental data distributions are very close to Keepin’s expected values. So are the total counts. In some time periods, the experimental data are significant difference from Keepin’s data. In both modes, the fitted data from the RR algorithm are not significantly different from Keepin’s values, indicating that the experimental data are close to Keepin’s data. They are all different from ENDF/B-VII.0 data from the plots in Figures 6.8 and 6.9. Our fit of the data to Keepin’s although not perfect, it is better than the difference between Keepin’s and ENDF/B-VII.0 data. In both modes, the variances in the later counting period are significantly different. This may come from the low counts in the later period. Another reason is that this is just from a single experimental data. To improve this, several detectors should be used to run several experiments to reduce the statistical error.

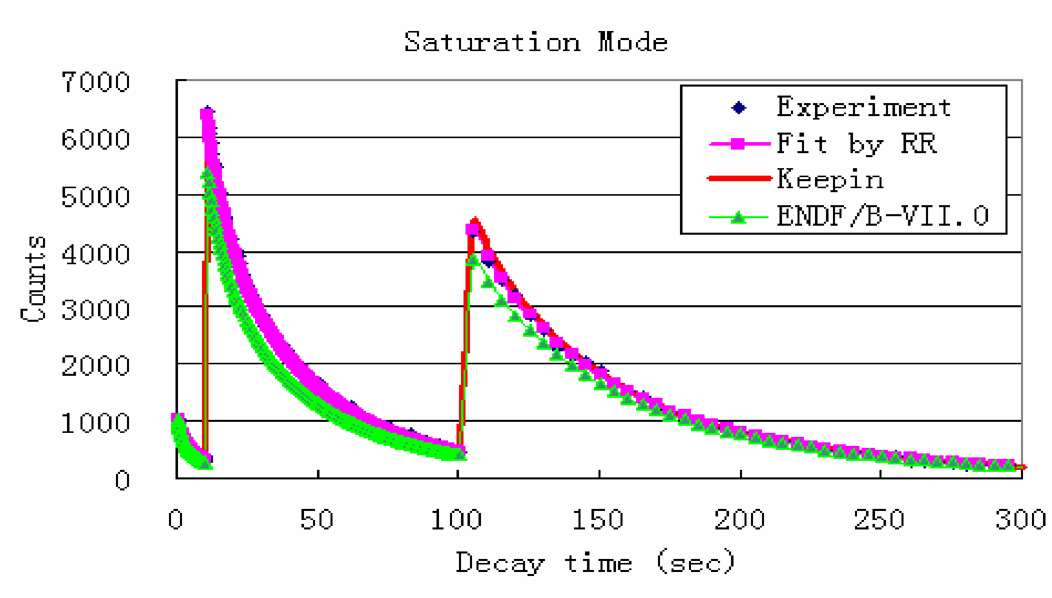


Figure 6.8 Experimental Data (Saturation Mode)

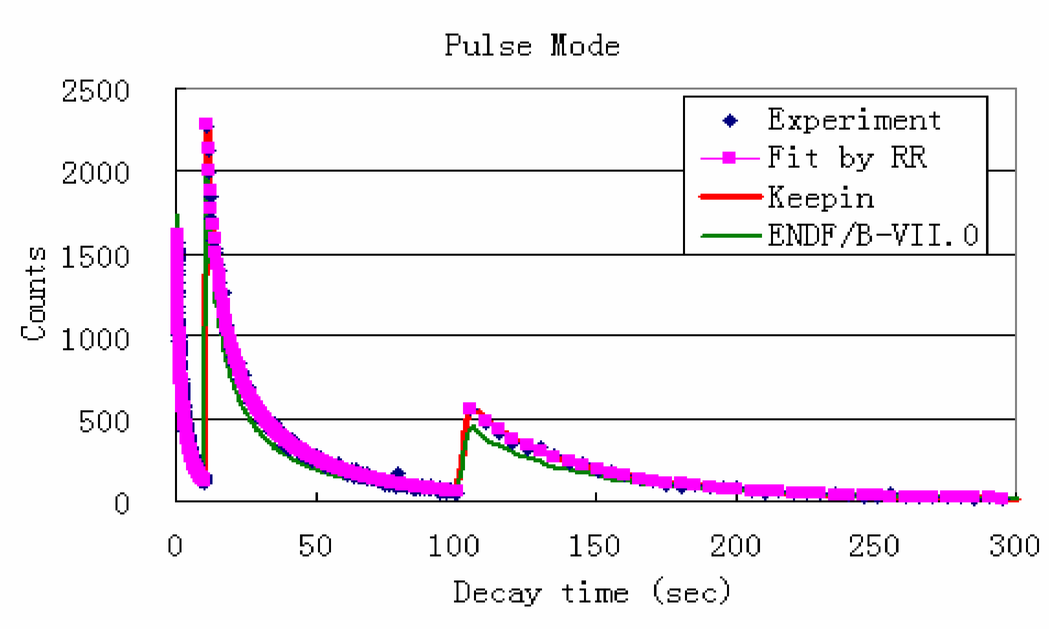


Figure 6.9 Experimental Data (Pulse Mode)

Table 6.17 Experimental Data Comparison (Saturation Mode)

| Time (s) | $\chi^2$ (Exp~Keepin) | $\chi^2$ (Fit~Keepin) | n | $\chi_0{}^2$(0.05,n-1) | $N_1$ (Exp) | $N_2$ (Keepin) | $N_1$-$N_2$ | 1.96σ |
|---|---|---|---|---|---|---|---|---|
| 0.443-1.0 | 31.65 | 0.42 | 22 | 32.67 | 21593 | 21326 | 267 | 288 |
| 1-2 | 35.85 | 0.57 | 40 | 55.76 | 32942 | 32999 | -57 | 356 |
| 2-4 | 50.42 | 2.42 | 80 | 101.88 | 52762 | 52171 | 591 | 450 |
| 4-6 | 88.86 | 2.37 | 80 | 101.88 | 41221 | 40732 | 489 | 398 |
| 6-8 | 91.22 | 1.77 | 80 | 101.88 | 33600 | 33478 | 122 | 359 |
| 8-10 | 66.83 | 1.39 | 80 | 101.88 | 28848 | 28520 | 328 | 333 |
| 10-40 | 58.83 | 3.14 | 60 | 79.08 | 219436 | 218924 | 512 | 918 |
| 40-80 | 105.43 | 16.20 | 80 | 101.88 | 102883 | 103846 | -963 | 629 |
| 80-300 | 161.37 | 55.40 | 80 | 101.88 | 72591 | 73465 | -874 | 528 |

Note:

1) Exp-Keepin: Comparison between experimental data and Keepin's expected data.
2) Fit-Keepin: Comparison between experimental data and Keepin's expected data.
3) $\chi_0{}^2$(0.05,n-1): Reference $\chi^2$ at significant level 0.05 and degrees of freedom n-1.
4) $N_1$, $N_2$: Total counts from experiment and Keepin's data in the given period.
5) 1.96σ: Reference value for significant difference with 95% confidence level.

Table 6.18 Experimental Data Comparison (Pulse Mode)

| Time (s) | χ2 (Exp~Keepin) | χ2 (Fit~Keepin) | n | $\chi_0{}^2$(0.05,n-1) | $N_1$ (Exp) | $N_2$ (Keepin) | $N_1$-$N_2$ | 1.96σ |
|---|---|---|---|---|---|---|---|---|
| 0.45-1.0 | 29.41 | 3.40 | 22 | 32.67 | 29384 | 29899 | 315 | 337 |
| 1-2 | 48.67 | 9.09 | 40 | 55.76 | 35513 | 35377 | 136 | 369 |
| 2-4 | 75.30 | 19.83 | 80 | 101.88 | 42498 | 42158 | 340 | 404 |
| 4-6 | 67.23 | 3.09 | 80 | 101.88 | 25319 | 25377 | -58 | 312 |
| 6-8 | 64.69 | 0.21 | 80 | 101.88 | 16800 | 16851 | -51 | 254 |
| 8-10 | 71.41 | 0.28 | 80 | 101.88 | 11630 | 11877 | -247 | 211 |
| 10-40 | 90.74 | 21.73 | 60 | 79.08 | 51906 | 51772 | 134 | 447 |
| 40-80 | 171.58 | 27.23 | 80 | 101.88 | 16084 | 15377 | 707 | 249 |
| 80-300 | 80.76 | 2.15 | 80 | 101.88 | 8337 | 8426 | -89 | 179 |

# 7. CONCLUSION

As reported in the literature, the fitted delayed neutron parameters agree very well with their measurement data, although these results differ markedly from one another. These differences are far larger than the small deviation they reported. The reported errors in their parameters are not correct actually. This project uses different numerical methods to investigate the estimation of delayed neutron decay parameters (relative yields and decay parameters) emitted from thermal fission of U-235.

Most researchers used the least squares method to obtain the fitted parameters. This project compares the least squares method (Matrix Inverse with SVD is used here) to other numerical methods, such as the Levenberg-Marquardt method, and the quasi-Newton method. To test these algorithms, data sets were made so that the Poisson distribution was the only variance. No experimental errors are included in the simulated data. The fitted results from different algorithms used on this relatively precise data revealed that all the fitted results can fit the simulated experimental data very well but the parameters may be totally different. The Matrix Inverse method without SVD and the quasi-Newton method cannot guarantee finding solutions for certain data sets because of ill-conditioned matrices in the algorithm and inappropriate searching methods. The Levenberg-Marquardt method can always find a solution, but its optimized result is not the best among those methods. Further investigation shows that the coefficient matrices (without regularization) in different algorithms are ill-posed causing the solution not to be unique. Due to the diversity of the regularization techniques, the fitted results have large variation. The optimized values from these methods are not global and the results are not stable.

An improved algorithm called Ridge Regression is used for the first time in this study to find delayed neutron parameters. A modified model considering the offset between the experimental data and the mathematical model is used to fit the measurement data in this project. To avoid an ill-conditioned problem, a regularization parameter h is introduced in the Ridge Regression algorithm and it guarantees finding a solution. Due to the regularization parameter, Ridge Regression introduces bias into the fitted results. The Ridge Regression method is to find a best “h” value between estimate variance and bias. Ridge Regression shows improvement in convergence and it achieves smaller sum of weighted square residuals, $E^2$, than other algorithms. Compared to the traditional least squares method, the Ridge Regression method achieves a smaller root of mean square error. This effect is very obvious to the short decay groups. This method is also not as sensitive to the initial guessed values as other algorithms.

The results with different numbers of fitted parameters are also investigated. It is found that reducing the number of unknown parameters cannot improve the optimization results. In saturation mode, the extracted values for long-lived groups are more stable. While for short-lived groups, they have large uncertainty. Pulse mode reduces the uncertainty for short-lived group parameters, but increases uncertainty for long decay groups. Further investigation found that the root of mean square error (RMSE) from such a precise Poisson distributed data still has extensive uncertainty some groups. The RMSE values are not as small as those claimed by previous researchers.

Many researchers compared the root of mean square error of the parameters to judge the fit. Actually, it means nothing due to the instability of the solution. Although the fitted parameters have large differences compared to Keepin's values, it does not mean the fitted parameters are not satisfactory. In fact, they are a good fit, as illustrated by very small sums of weighted squared residuals. The comparison shows that the fitted data sets from RR are not significantly different from Keepin's theoretical data sets. However, for a small change in data set (generated by increasing $A_6$ by 10%), the newly fitted data set resulting from this small change can be significantly different from Keepin's expected values at a 95% confidence level. With the increase of flight time and/or decrease of initial count rate, the influence of the change in the counts from short decay groups becomes less and less significant on the fitted result. But if the flight time is too long, this small change in the counts will no longer have an effect on the fitted values, no matter how high the initial count rate because the delayed neutrons from the shortest-lived group have decayed away. The fraction of delayed neutrons from the shortest-lived group is much higher in pulse mode than in saturation mode for the same decay time with the same initial count rates. Thus, the change from short decay group is more sensitive to the fitted results in pulse mode and its influence lasts much longer than in saturation mode.

Finally, based on the experimental data sets from saturation and pulse modes, the optimization result from the RR algorithm is the best among these four algorithms (MI-SVD, LM, and QN). It is the same for experimental data sets. The fitted values are satisfactory with Keepin's expected values, but they obviously differ from ENDF/B-

VII.0 data sets. The experimental data sets are a little different from Keepin’s expected values in some periods under a 95% confidence level. This difference may be from the low count rate in the experiment and a single experiment data set. It suggests using several detectors and running multiple experiments to reduce the statistical error.

# REFERENCES

[1] A. D'Angelo, Progress in Nuclear Energy, **41**, 1-4, 5-38 (2002).

[2] R. B. Roberts, R. C. Meyer, L. R. Hafstad, and P. Wang, Phys. Rev., **55**, 664 (1939).

[3] N. Bohr and J. A. Wheeler, Phys. Rev., **55**, 426 (1939).

[4] A. H. Snell, V. A. Nedzel, H. W. Ibser, J. S. Levinger, R. G. Wilkinson, M. B. Sampson, Phys. Rev., **72**, 541 (1947).

[5] W. C. Redman and D. Saxon, Phys. Rev., **72**, 570 (1947).

[6] D. J. Hughes, J. Dabbs, A. Cahn, and D. Hall, Phys. Rev., **73**, 111 (1948).

[7] G. R. Keepin, T. F. Wimett, and R. K. Zeigler, Phys. Rev. **107**, 1044 (1957).

[8] G. R. Keepin, T. F. Wimett, and R. K. Zeigler, J. Nuclear Energy, **6**, 1 (1957).

[9] G. R. Keepin, Phys. Rev., **106**, 6 (1957).

[10] G. R. Keepin, Physics of Nuclear Kinetics, (Addison Wesley Publishing Company, Reading, MA, 187, 1965).

[11] S. Cox, P. Fields, A. Friedman, R. Sjoblom, and A. Smith, Phys. Rev. **112**, 960 (1958).

[12] C. F. Masters, M. M. Thorpe, and D. B. Smith, Nucl. Sci. Eng., **36**, 202-208 (1969).

[13] M. S. Krick and A. E. Evans, Nucl. Sci. Eng., **47**, 311 (1972).

[14] D. R. Alexander and M. S. Krick, Nucl. Sci. Eng., **62**, 627-635 (1977).

[15] J. T. Caldwell, E. J. Dowdy, and G. M. Worth, Los Alamos Report LA-UR-73-

968 (1973).

[16] R. J. Tuttle, Nucl. Sci. Eng. **56**, 37 (1975).

[17] G. Rudstam, Nucl. Sci. Eng. **80**, 238 (1982).

[18] T. R. England, M. C. Brady, W. B. Wilson, R. E. Schenter, R, F. M. Mann, Proceedings of International Conference on Nuclear Data for Basic and Applied Science, (Santa Fe, New Mexico, Gordon and Breach Publishers, May 13-17, 1985).

[19] M. C. Brady and T. R. England, Nucl. Sci. and Eng., **103**, 129 (1989).

[20] R. W. Waldo, R. A. Karam, and R. A. Meyer, Phys. Review C, **23**, 3, 1113-1127 (Mar. 1981).

[21] R. W. Waldo, Ph.D dissertation, Georgia Institute of Technology, (May, 1980).

[22] G. Benedetti, A. Cesana, V. Sangiust, M. Terrani, and G. Sandrelli, Nucl Sci. Eng., **80**, 379-387 (1982).

[23] Y. Kaneko, F. Akino, and T. Yamane, J. Nucl. Sci. Technol., **25**, 9, 673-681 (Sept., 1988).

[24] M. F. Villani, G. P. Couchell, M. H. Haghighi, D. J. Pullen, W. A. Schier and Q. Sharfuddin, Nucl. Sci. Eng., **111**, 4, 422-432, (August, 1992).

[25] H. H. Saleh, T. A. Parish, Nucl. Sci. Eng., **125**, 51-60 (1997).

[26] W. S. Charlton, T. A. Parish, S. A. Raman, N. Shinohara, M. Andoh, Proceedings of International Conference on Nuclear Data for Science and Technology, Trieste, Conference Proceedings edited for the Italian Physical Society, **59**, 491 (19-24 May, 1997).

[27] W. S. Charlton, T. A. Parish, S. A. Raman, Proceedings of International Conference on Nuclear Data for Science and Technology, Oct. 5-8, 1998, (Islandia Marriot Long Island Happurage, New York, **1**, 190, 1998).

[28] W. S. Charlton, Master Thesis, Texas A&M University (August, 1998).

[29] D. Loaiza, G. Brunson, R. Sanchez, K. Butterfield, Los Alamos National Laboratory Report, LA-UR-97-1858 (1997).

[30] D. J. Loaiza, Nuclear Instruments and Methods in Physics Research, **A422**, 43-46 (1999).

[31] J. Blachot, C. Chung and F. Storrer, Ann. Nucl. Energy, **24**, 6, 489-504 (1997).

[32] G. D. Spriggs, J. Campbell and V. M. Piksaikin, Los Alamos Technical Report LA-UR-98-1619 (1998).

[33] D. J. Loaiza and F. E. Haskin, Proceedings of International Conference on Physics of Nuclear Science and Technology, Islandia Marriot Long Island Happurage, New York, **1**, 303 (Oct, 1998).

[34] G. D. Spriggs, J. M. Campbell, Progress in Nuclear Energy, **41**, 1-4, 145-201 (2002).

[35] V. M. Piksaikin, L. E. Kazakov, S. G. Isaev, M. Z. Tarasko, V. A. Roshchenko, R. G. Tertytchnyi, G. D. Spriggs, J. M. Campbell, Progress in Nuclear Energy, **41**, 1-4, 203-222 (2002).

[36] V. M. Piksaikin, S. G. Isaev and A. A. Goverdovski, Progress in Nuclear Energy, **41**, 1-4, 361-384 (2002).

[37] S. B. Borzakov, TS. Panteleev, S. S. Pavlov, I. N. Ruskov, and Yu. S.

Zamyatnin, Progress in Nuclear Energy, **41**, 1-4, 125-133 (2002).

[38] W. B. Wilson and T. R. England, Progress in Nuclear Energy, **41**, 1-4, 71-107 (2002).

[39] B. Pfeiffer, K. L. Kratz, and P. Moller, Progress in Nuclear Energy, **41**, 1-4, 39-69 (2002).

[40] V. M. Piksaikin, L. E. Kazakov, G. G. Korolev, S. G. Isaev, V. A. Roshchenko, and M. Z. Tarasko, Atomic Energy, **92**, 2, 147 (2002).

[41] V. M. Piksaikin, L. E. Kazakov, V. A. Roshchenko, S. G. Isaev, G. G. Korolev, A. A. Goverdovski, R. G. Tertytchnyi, Atomic Energy, **92**, 3, 254 (2002).

[42] S. G. Isaev, V. M. Piksaikin, L. E. Kazakov, M. Z. Tarasko, Proc. of XIV-th Int. Workshop on Nuclear Fission Physics, 257-264 (October 12-15, 1998).

[43] V. M. Piksaikin, Y. F. Balakshev, S. G. Isaev, L. E. Kazakov, G. G. Korolev, and V. I. Mil’shin, Atomic Energy, **85**, 1 (1998).

[44] D. Dore, J-C.David, M-L. Giacri, J-M. Laborie, X. Ledoux, M. Petit, D. Ridikas and A. Vanlauwe, Journal of Physics: Conference Series **41**, 241–247 (2006).

[45] D. Dore, J-C.David, V. Macary, J-M. Laborie, X. Ledoux, D. Ridikas and A. Vanlauwe, Proceedings of International Conference on Nuclear Data for Science and Technology, 1269-1272 (2007).

[46] V. M. Piksaikin, V. A. Roshchenko, and G. G. Korolev, Atomic Energy, Vol. **102**, 2, 151-159 (2007).

[47] A. Dos Santos, R. Diniz, R. Jerez, L. Antonio Mai, M. Yamaguchi, Annals of Nuclear Energy, **33**, 917-923 (2006).

[48] B. Geslot, C. Jammes, B. Gall, Annals of Nuclear Energy, **34**, 652-660 (2007).

[49] Evaluated Nuclear Data File (ENDF), http://www-nds.iaea.org/exfor/endf.htm, Database Version of May 14, 2008.

[50] Reactor Theory (Neutron Characteristics), DOE-HDBK-1019/1-93, Rec.0, 32, www.tpub.com/content/doe/h1019v1/css/h1019v1_136.htm (2007).

[51] W. H. Press, B. P. Flannery, S. A. Teukolsky, W. T. Vetterling, Numerical Recipes: The Art of Scientific Computing, 1 Edition, (Cambridge University Press, New York, **14**, 498-539, 1986).

[52] D. Marquardt, SIAM J. Appl. Math., **11**, 431-441 (1963).

[53] K. Levenberg, Quart. Appl. Math., **2**, 164-168 (1944).

[54] J. E. Dennis Jr, J. J. More, SIAM Review, **19**, 46-89 (1977).

[55] IMSL MATH/LIBRARY, chapter 8: Optimization, 886-912 (1997).

[56] A. N. Tikhonov, V. Y. Arsenin, Solutions of Ill-posed Problem, (John Wiley & Sons, Washington D.C., 1977).

[57] A. E. Hoerl and R. N. Kennard, Technometrics, **12**, 1, 55-67 (1970).

[58] P. C. Hansen, Rank-deficient and Discrete Ill-posed Problems: Numerical Aspects of Linear Inversion, 102-240 (Society for Industrial and Applied Mathematics, Philadelphia, PA, 1998).

[59] G. H. Golub, M. Heath and G. Wahba, Technometrics **21**, 215–223 (1979).

[60] P. L. Xu, Geophys. J. Int., **135**, 505-514 (1998).

[61] G. Wahba, Spline Models for Observational Data, CBMS-NSF Regional Conference Series in Applied Mathematics, **59**, 102-150 (Society for Industrial

and Applied Mathematics, Philadelphia, PA, 1990).

[62] A. Heinrich, Master Thesis, Texas A&M University (May, 2008).

[63] D. J. Loaiza, F. E. Haskin, Los Alamos Technical Report, LA-UR-97-2252, (1997).

[64] J. E. Turner, Atoms, Radiation, and Radiation Protection, 3rd version, (WILEY-VCH Verlag GmbH & Co. KgaA, Weinhein, 327, 2007).

# APPENDIX A

# PROGRAM CODES

### A.1 Matrix Inverse Algorithm

```
!*********** Matrix Inverse with SVD *********************************
      USE NUMERICAL_LIBRARIES
      INTEGER I, J, L, K, M, NN, COUNT, ISEED
      INTEGER IPATH, LDA, NRA, NCA, LDU, LDV, NT
      REAL*8 A0(6),R0(6),A(6),R(6),H(6,6),G(6,6),F,    DT,TOL
      REAL*8 UH(6,6),UG(6,6),VH(6,6),VG(6,6),SH(6),SG(6)
      REAL*8 SHMIN, SHMAX, SGMIN, SGMAX, CVM_H(6,6), CVM_G(6,6)
      PARAMETER (NN=6,NT=620, IPATH=11,NRA=6,NCA=6)
      PARAMETER (LDA=NRA, LDU=NRA, LDV=NCA)
      REAL*8 D(6), E(6), W(NT), D_A(6), D_R(6), DWELL(NT)
      REAL*8 THEORY(NT), MEASURED(NT), STEP_TIME(NT), TEMP(NT),
      REAL*8 CT(NT), T
C     COMMON MEASURED, STEP_TIME, THEORY, A0, R0, DWELL, COUNT

      DATA A0/3.3E-2, 2.19E-1, 1.96E-1, 3.95E-1, 1.15E-1, 4.2E-2/
      DATA R0/1.24E-2, 3.05E-2, 1.11E-1, 3.01E-1, 1.14E0, 3.01E0/
      TOL=10.*AMACH(6)
      COUNT=4.0E5
      DT=0.0

      DO I=1,400
              STEP_TIME(I) = (REAL(I)-0.5)*0.025
              DWELL(I)=0.025
      ENDDO
      DO I=401,580
              STEP_TIME(I)=10.0+(REAL(I-400)-0.5)*0.5
              DWELL(I)=0.5
      ENDDO
      DO I=581,NT
              STEP_TIME(I) = 100.0+(REAL(I-580)-0.5)*5.0
              DWELL(I)=5.0
      ENDDO
      ISEED = -1031501663
      DO I=1,NT
        T=STEP_TIME(I)+DT
        THEORY(I)= COUNT*DWELL(I)*
  &   (A0(1)*EXP(-R0(1)*T) + A0(2)*EXP(-R0(2)*T)
  &  + A0(3)*EXP(-R0(3)*T) + A0(4)*EXP(-R0(4)*T)
  &  + A0(5)*EXP(-R0(5)*T) + A0(6)*EXP(-R0(6)*T))
        RAND=RAN(ISEED)
      MEASURED(I)=INT(THEORY(I)+SQRT(THEORY(I))*ANORIN(RAND)+0.5)
      ENDDO
      DO J=1,6
            D_A(J)=0.
            A(J)=A0(J)
            D_R(J)=0.
            R(J)=R0(J)
      ENDDO
      DO K = 1,10
            DO J=1,6
                  D(J)=0.
                  E(J)=0.
                        DO M =1,6
                              H(J,M)=0.
                              G(J,M)=0.
                        ENDDO
            ENDDO
            F=0.

C     *********Define matrices: H(n,m),D(n),E(n,m),G(n)***************
            DO I=1,NT
              T=STEP_TIME(I)+DT
              TEMP(I)= COUNT*DWELL(I)*(A(1)*EXP(-R(1)*T)
  &         + A(2)*EXP(-R(2)*T)
  &         + A(3)*EXP(-R(3)*T)
  &         + A(5)*EXP(-R(5)*T)
  &         + A(4)*EXP(-R(4)*T)
  &         + A(6)*EXP(-R(6)*T))
              W(I)=1.0/MEASURED(I)
                  DO L=1,6
                  D(L)=D(L)+W(I)*EXP(-R(L)*T)*(MEASURED(I)-TEMP(I))
            E(L)=E(L)+W(I)*(MEASURED(I)-TEMP(I))*T*EXP(-R(L)*t)* A(L)
                        DO M=1,6
                  H(L,M)=H(L,M)+W(I)*EXP(-R(L)*T)*EXP(-
                  R(M)*T)*DWELL(I)*COUNT
          G(L,M)=G(L,M)+W(I)*T*EXP(-R(L)*T)*EXP(-R(M)*T)*(-A(M))*T*A(L)
  &        *DWELL(I)*COUNT
                        ENDDO
                  ENDDO
            ENDDO

C   *******Inverse of matrix using singular value decomposition ******
```

```
          CALL
      DLSVRR(NRA,NCA,H,LDA,IPATH,TOL,IRANK,SH,UH,LDU,VH,LDV)
          CALL
      DLSVRR(NRA,NCA,G,LDA,IPATH,TOL,IRANK,SG,UG,LDU,VG,LDV)
          SHMAX=0.
          SGMAX=0.
          DO J=1,6
              IF(SH(J).GT.SHMAX) SHMAX=SH(J)
              IF(SG(J).GT.SGMAX) SGMAX=SG(J)
          ENDDO
          SHMIN=SHMAX*1.0E-6
          SGMIN=SGMAX*1.0E-6
          DO J=1,6
              IF(SH(J).LT.SHMIN) SH(J)=0.
              IF(SG(J).LT.SGMIN) SG(J)=0.
          ENDDO
          CALL SVBKSB(UH,SH,VH,NN,NN,NN,NN,D,D_A)
          CALL SVBKSB(UG,SG,VG,NN,NN,NN,NN,E,D_R)
          CALL SVDVAR(VH,NN,NN,SH,CVM_H,NN)
          CALL SVDVAR(VG,NN,NN,SG,CVM_G,NN)
          DO J=1,6
              A(J)=A(J)+0.8*D_A(J)
              R(J)=R(J)+0.8*D_R(J)
          ENDDO
      ENDDO

      DO I=1,NT
              T=STEP_TIME(I)+DT
              TEMP(I)= COUNT*DWELL(I)*(A(1)*EXP(-R(1)*T)
   &      + A(2)*EXP(-R(2)*T)
   &      + A(3)*EXP(-R(3)*T)
   &      + A(4)*EXP(-R(4)*T)
   &      + A(5)*EXP(-R(5)*T)
   &      + A(6)*EXP(-R(6)*T))

              F=F+(MEASURED(I)-TEMP(I))**2/MEASURED(I)
      ENDDO
C     ****** Complete one run ************************************
          WRITE (6,901) (A(L)/A0(L), L=1,6)
          WRITE (6,902) (R(L)/R0(L), L=1,6)
          WRITE (6,903) F
901       FORMAT("THE YIELD RATIO IS: ", //,6(F12.5,/))
902       FORMAT("THE LAMMDA RATIO IS: ", //,6(F12.5,/))
903       FORMAT (" THE SQUARE DIFFERENCE F=", F12.5)
```

```
                DO I=1,620
                WRITE (6,'(F8.3, 3(3X,F10.2))') STEP_TIME(I), THEORY(I),
   &    MEASURED(I),TEMP(I)
        IF (I.EQ.400) THEN
                        PAUSE
                ENDIF
                ENDDO
        END

C
        *********************SUBROUTINE*****************************
***
C   ******************** Singular Value Decomposition *******************
        SUBROUTINE SVDVAR(V,MA,NP,W,CVM,NCVM)
        PARAMETER (MMAX=20)
        REAL*8 V(NP,NP),W(NP),CVM(NCVM,NCVM),WTI(MMAX)
        DO 11 I=1,MA
          WTI(I)=0.0
          IF(W(I).NE.0.0) WTI(I)=1.0/(W(I)*W(I))
11      CONTINUE
        DO 14 I=1,MA
          DO 13 J=1,I
            SUM=0.0
            DO 12 K=1,MA
              SUM=SUM+V(I,K)*V(J,K)*WTI(K)
12          CONTINUE
         CVM(I,J)=SUM
         CVM(J,I)=SUM
13       CONTINUE
14      CONTINUE
        RETURN
        END

C        ******************** Subroutine SVBKSB
*******************************
C        ***** Solves A*X=B for a vector X. Where, X=V*[diag(1/wj)]*(U'.B) ***
        SUBROUTINE SVBKSB(U,W,V,M,N,MP,NP,B,X)
        PARAMETER (NMAX=20)
        REAL*8 U(MP,NP),W(NP),V(NP,NP),B(MP),X(NP),TMP(NMAX)

        DO J=1,N
        S=0.
                IF(W(J).NE.0.) THEN
                        DO I=1,M
```

```
                  S=S+U(I,J)*B(I)
            ENDDO
            S=S/W(J)
      ENDIF
TMP(J)=S
ENDDO
DO J=1,N
      S=0.
      DO JJ=1,N
            S=S+V(J,JJ)*TMP(JJ)
      ENDDO
      X(J)=S
ENDDO
RETURN
END
```

## A.2 Levenberg-Marquardt Algorithm

```
!********** Levenberg-Marquardt Algorithm ****************************
      USE NUMERICAL_LIBRARIES
      INTEGER N, I, K, M
      INTEGER LDFJAC
      INTEGER IPARAM(7),L, NOUT
      PARAMETER (LDFJAC=620,NT=620,M=620,N=13)
      REAL*8 MEASURED(NT),STEP_TIME(NT),THEORY(NT),DWELL(NT),F
      REAL*8 RPARAM(7), X(13),VALUES(13), XSCALE(13),TF(NT),
      INTEGER*4 ISEED
      EXTERNAL FCN
      REAL*8 FVEC(M),FJAC(M,N),FSCALE(M),TEMP(NT), XGUESS(13)
      COMMON MEASURED, STEP_TIME, THEORY, VALUES, DWELL
      DATA VALUES/3.3E-2, 2.19E-1, 1.96E-1, 3.95E-1, 1.15E-1, 4.2E-2,
     &  1.24E-2, 3.05E-2, 1.11E-1, 3.01E-1, 1.14E0, 3.01E0, 4.0E5/

      DO I=1,13
            XGUESS(I)=1.0
      ENDDO
      DO I=1,400
            STEP_TIME(I) = (REAL(I)-0.5)*0.025
            DWELL(I)=0.025
      ENDDO
      DO I=401,580
            STEP_TIME(I)=10.0+(REAL(I-400)-0.5)*0.5
            DWELL(I)=0.5
      ENDDO
      DO I=581,NT
            STEP_TIME(I) = 100.0+(REAL(I-580)-0.5)*5.0
            DWELL(I)=5.0
      ENDDO
      DO I=1,13
            XSCALE(I) =0.05
      ENDDO
      ISEED =-10
      DO K=1,1
            CALL DISTRIBUTE(VALUES,ISEED)

C     ********** CHANGE DIFFERENT ISEED FOR EACH LOOP  **********
            F=0
            IPARAM(1) = 1
            IPARAM(3) = 200000
```

```
                IPARAM(4) = 200000
                IPARAM(5) = 200000
                IPARAM(6) = 1
                FSCALE = 0.005
                CALL DU4LSF (IPARAM,RPARAM)
                RPARAM(4)=2.0*RPARAM(4)
                CALL DUNLSF (FCN,M,N,XGUESS,XSCALE,FSCALE,IPARAM,
    &           RPARAM,X,FVEC,FJAC,LDFJAC)
                CALL UMACH (2,NOUT)
                F=0
                DO I=1, NT
                T=STEP_TIME(I)
                TEMP(I)= X(13)*VALUES(13)*DWELL(I)*
    &          (X(1)*VALUES(1)*EXP(-X(7)*VALUES(7)*T)
    &        +   X(2)*VALUES(2)*EXP(-X(8)*VALUES(8)*T)
    &        +   X(3)*VALUES(3)*EXP(-X(9)*VALUES(9)*T)
    &        +   X(4)*VALUES(4)*EXP(-X(10)*VALUES(10)*T)
    &        +   X(5)*VALUES(5)*EXP(-X(11)*VALUES(11)*T)
    &        +   X(6)*VALUES(6)*EXP(-X(12)*VALUES(12)*T))
                F=F+(MEASURED(I)-TEMP(I))**2/MEASURED(I)
                ENDDO
                WRITE (6,901) (X(L), L=1,13)
901             FORMAT("THE SOLUTION IS ", //,13(F12.5,/))
                WRITE (*,*) F
        ENDDO
        DO I=1,NT
                WRITE(*,903) STEP_TIME(I), THEORY(I), MEASURED(I),TEMP(I)
903             FORMAT(3X,F8.3,3(3X,F12.2))
                IF (I.EQ.200) THEN
                PAUSE
                ENDIF
        ENDDO
        END

C       *********************SUBROUTINE*****************************
C       fills the common block with values for measured and theory
        SUBROUTINE DISTRIBUTE(Y,ISEED)
        PARAMETER (NT=620)
        REAL*8 THEORY(NT), VALUES(13)
        REAL*8 T, Y(13), STEP_TIME(NT), MEASURED(NT), DWELL(NT)
        REAL*4 RAND
        INTEGER N, I
        INTEGER*4 ISEED
        COMMON MEASURED, STEP_TIME, THEORY, VALUES, DWELL
```

```
      DO I=1,NT
              T=STEP_TIME(I)
              THEORY(I)= Y(13)*DWELL(I)*
 &     (Y(1)*EXP(-Y(7)*T) + Y(2)*EXP(-Y(8)*T)
 &    + Y(3)*EXP(-Y(9)*T) + Y(4)*EXP(-Y(10)*T)
 &    + Y(5)*EXP(-Y(11)*T)+ Y(6)*EXP(-Y(12)*T))
              RAND=RAN(ISEED)
     MEASURED(I)=INT(THEORY(I)+SQRT(THEORY(I))*ANORIN(RAND)+0.5)

      ENDDO
      RETURN
      END

C     *********************DEFINE FUNCTION**********************
      SUBROUTINE FCN(J,N,X,TF)
      PARAMETER (NT=620)
      INTEGER N,I,J
      REAL*8 MEASURED(NT),STEP_TIME(NT),THEORY(NT),VALUES(13)
      REAL*8 X(13),TF(NT),TEMP(NT),DWELL(NT),T
      COMMON MEASURED, STEP_TIME, THEORY, VALUES, DWELL

      DO I=1,NT
              T=STEP_TIME(I)
              TEMP(I)= X(13)*VALUES(13)*DWELL(I)*
 &           (X(1)*VALUES(1)*EXP(-X(7)*VALUES(7)*T)
 &        +   X(2)*VALUES(2)*EXP(-X(8)*VALUES(8)*T)
 &        +   X(3)*VALUES(3)*EXP(-X(9)*VALUES(9)*T)
 &        +   X(4)*VALUES(4)*EXP(-X(10)*VALUES(10)*T)
 &        +   X(5)*VALUES(5)*EXP(-X(11)*VALUES(11)*T)
 &        +   X(6)*VALUES(6)*EXP(-X(12)*VALUES(12)*T))
              TF(I)=(MEASURED(I)-TEMP(I))/SQRT(MEASURED(I))
      ENDDO
      RETURN
      END
```

## A.3 Quasi-Newton Algorithm

```
!********** Quasi-Newton Algorithm ********************************
        USE NUMERICAL_LIBRARIES
        INTEGER N, I
        INTEGER IPARAM(7),L, NOUT
        PARAMETER (N=13, NT=620)
        REAL*8 MEASURED(NT), STEP_TIME(NT), THEORY(NT),F,DWELL(NT)
        REAL*8 FSCALE, RPARAM(7), X(13),VALUES(13), XSCALE(13), Y(13)
        REAL*8 T, TEMP, F_1
        INTEGER*4 ISEED
        EXTERNAL DIFFSQUARES
        COMMON MEASURED, STEP_TIME, THEORY, VALUES, DWELL
        DATA VALUES/3.3E-2, 2.19E-1, 1.96E-1, 3.95E-1, 1.15E-1, 4.2E-2,
  &    1.24E-2, 3.05E-2, 1.11E-1, 3.01E-1, 1.14E0, 3.01E0, 4.0E5/

        DO I=1,400
                STEP_TIME(I) = (REAL(I)-0.5)*0.025
                DWELL(I) = 0.025
        ENDDO
        DO I=401,580
                STEP_TIME(I)=10.0+(REAL(I-400)-0.5)*0.5
                DWELL(I) = 0.5
        ENDDO
        DO I=581,NT
                STEP_TIME(I) = 100.0+(REAL(I-580)-0.5)*5.0
                DWELL(I) = 5.0
        ENDDO
        DO I=1,13
                Y(I) = 1.0
                XSCALE(I) =0.5
        ENDDO
        ISEED =774385225
                CALL DISTRIBUTE(VALUES,ISEED)
                IPARAM(1) = 1
                IPARAM(3) = 20000
                IPARAM(4) = 20000
                IPARAM(5) = 20000
                IPARAM(6) = 1
                FSCALE =0.05
                CALL DIFFSQUARES(N,Y, F_1)
                CALL DUMINF(DIFFSQUARES,N,Y, XSCALE, FSCALE,IPARAM,
                RPARAM,X,F)
                RPARAM(1)=2.0*RPARAM(1)
                CALL UMACH (2, NOUT)
                WRITE (6,901) (X(L), L=1,13)
901             FORMAT("THE SOLUTION IS ", //,13(F12.5,/))
                WRITE(6,*) F
        DO I=1,NT
                T=STEP_TIME(I)
                TEMP= X(13)*VALUES(13)*DWELL(I)*
    &                   (X(1)*VALUES(1)*EXP(-X(7)*VALUES(7)*T)
    &            + X(2)*VALUES(2)*EXP(-X(8)*VALUES(8)*T)
    &            + X(3)*VALUES(3)*EXP(-X(9)*VALUES(9)*T)
    &            + X(4)*VALUES(4)*EXP(-X(10)*VALUES(10)*T)
    &            + X(5)*VALUES(5)*EXP(-X(11)*VALUES(11)*T)
    &            + X(6)*VALUES(6)*EXP(-X(12)*VALUES(12)*T))
           WRITE (6,'(F8.3,3(1F12.2))') STEP_TIME(I), THEORY(I),
    &     MEASURED(I),TEMP
                IF (I.EQ.200) THEN
                        PAUSE
                ENDIF
        ENDDO
        END

C       **********************SUBROUTINE*****************************
        SUBROUTINE DISTRIBUTE(Y,ISEED)
        PARAMETER (NT=620)
        REAL*8 THEORY(NT),VALUES(13) ,MEASURED(NT)
        REAL*8 T, Y(13),STEP_TIME(NT),DWELL(NT)
        REAL*4 RAND
        INTEGER N, I
        INTEGER*4 ISEED
        COMMON MEASURED, STEP_TIME, THEORY, VALUES, DWELL
        DO I=1,NT
                T=STEP_TIME(I)
                THEORY(I)= Y(13)*DWELL(I)*(Y(1)*EXP(-Y(7)*T)
    &          + Y(2)*EXP(-Y(8)*T) + Y(3)*EXP(-Y(9)*T)
    &          + Y(4)*EXP(-Y(10)*T) + Y(5)*EXP(-Y(11)*T)
    &          + Y(6)*EXP(-Y(12)*T))
                RAND=RAN(ISEED)
       MEASURED(I)=INT(THEORY(I)+SQRT(THEORY(I))*ANORIN(RAND)+0.5)
        ENDDO
        RETURN
        END

C       *********************DEFINE FUNCTION**********************
```

```
    SUBROUTINE DIFFSQUARES(N,X,F)
    INTEGER N, I
    PARAMETER (NT=620)
    REAL*8 MEASURED(NT),STEP_TIME(NT),THEORY(NT),VALUES(13)
    REAL*8 X(13),F,TEMP,TEMP2,T,DWELL(NT)
    COMMON MEASURED, STEP_TIME, THEORY, VALUES, DWELL
           F=0.0
           DO I=1,NT
           T=STEP_TIME(I)
           TEMP= X(13)*VALUES(13)*DWELL(I)*
&            (X(1)*VALUES(1)*EXP(-X(7)*VALUES(7)*T)
&        + X(2)*VALUES(2)*EXP(-X(8)*VALUES(8)*T)
&        + X(3)*VALUES(3)*EXP(-X(9)*VALUES(9)*T)
&        + X(4)*VALUES(4)*EXP(-X(10)*VALUES(10)*T)
&        + X(5)*VALUES(5)*EXP(-X(11)*VALUES(11)*T)
&        + X(6)*VALUES(6)*EXP(-X(12)*VALUES(12)*T))
           F=F+(MEASURED(I)-TEMP)**2/MEASURED(I)
           ENDDO
    RETURN
    END
```

## A.4 Ridge Regression Algorithm

```
% ****  Ridge Regression for saturation irradiation mode  *****
clear all;
format short e
a10=0.033;a20=0.219;a30=0.196;a40=0.395;a50=0.115;a60=0.042;
r10=0.0124;r20=0.0305;r30=0.111;r40=0.301;r50=1.14;r60=3.01;
A0=[a10; a20; a30; a40; a50; a60];
R0=[r10; r20; r30; r40; r50; r60];

% *** Define the initial guessed values, such as records number and
%  flight time, background noise b, original counts, etc. ***********
b=1.0;
v0=[A0; R0;b];
count=4.0e5;
dt=0.0;
for i=1:13
   x0(i)=1.0;
end
x=x0';

%  To read the measured counts  ***************
[ti, keepin, measured, fit]=textread('p10.txt');
rn=size(measured,1);

%  For saturation mode, call [f,J,temp]=LMfun13noiseRATIO(x)
%  For pulse mode, call [f,J,temp]=LMfun13noiseRATIO(x);
%  It should be the same function name when it is called later.

[f,J,temp]=LMfun13noiseRATIO(x);
new_f=f;
old_f=700.0;
loop=0;
rx=size(x,1);

dn=dt/0.025;
for i=1:rn
   if i<(401-dn)
      t0(i)=(i-0.5)*0.025+dt;
      dwell(i)=0.025;
   elseif i<(581-dn)
      t0(i)=10.0+(i-400-0.5+dn)*0.5;
      dwell(i)=0.5;
   else
      t0(i)=100.0+(i-580-0.5+dn)*5.0;
      dwell(i)=5.0;
   end
   w(i)=1.0/measured(i);
end

while abs((new_f-old_f)/old_f)>=1.0e-4
   [f,J,temp]=LMfun13noiseRATIO(x);
   old_f=f;
   for i=1:rn
      delta_y(i,1)=sqrt(w(i))*(measured(i)-temp(i));
      end

%  *************** Ridge Regression method ************
%  If h is decided by GCV method, call this function:
%  [delta_x, rss, delta_x_ss, dof, h, G]=ridge(delta_y, J);

%  If h is given by L-Curve, call this function:
h=1.2;
[delta_x,rss,delta_x_ss,dof]=ridge(delta_y, J, h);
%  ********* end of Ridge Regression ****************

   for i=1:12
      x(i)=x(i)+delta_x(i);
   end
   x(13)=delta_x(13);
   [f,J,temp]=LMfun13noiseRATIO(x);
   new_f=f;
   loop=loop+1;
end
%  End "while" loop.

% ********** Calculate Root of mean square error  ********
[f, J, temp]=LMfun13noiseRATIO(x);
chi_square=(J*delta_x-delta_y)'*(J*delta_x-delta_y)+h*h*delta_x'*delta_x;
f0=(J*delta_x-delta_y)'*(J*delta_x-delta_y);
AA=inv(J'*J+h*h*eye(rx))*J';
VAR1=f/(rn-rx)*diag(AA*AA');
BIAS1=-h*h*inv(J'*J+h*h*eye(rx))*x;
RMSE=sqrt(VAR1+BIAS1.*BIAS1).*v0;


% *******  Define the function for saturation mode *********
function [f, J, temp]=LMfun13noiseRATIO(x)
count=4.0e5;
dt=0.0;
a10=0.033;a20=0.219;a30=0.196;a40=0.395;a50=0.115;a60=0.042;
r10=0.0124;r20=0.0305;r30=0.111;r40=0.301;r50=1.14;r60=3.01;
b=1.0;
va0=[a10; a20; a30; a40; a50; a60];
vr0=[r10; r20; r30; r40; r50; r60];
```

```
v0=[va0; vr0; b];
f=0.0;

[ti, keepin, measured, fit]=textread ('p10.txt');
[rn]=size(measured);
dn=dt/0.025;

for i=1:rn
   if i<(401-dn)
     t0(i)=(i-0.5)*0.025+dt;
     dwell(i)=0.025;
   elseif i<(581-dn)
     t0(i)=10.0+(i-400-0.5+dn)*0.5;
     dwell(i)=0.5;
   else
     t0(i)=100.0+(i-580-0.5+dn)*5.0;
     dwell(i)=5.0;
   end
   t=t0(i);
   w(i)=1.0/measured(i);
   temp(i)=count*dwell(i)*(x(1)*v0(1)*exp(-x(7)*v0(7)*t)+x(2)*v0(2)*exp(-x(8)*v0(8)*t)+…
                           x(3)*v0(3)*exp(-x(9)*v0(9)*t)+x(4)*v0(4)*exp(-x(10)*v0(10)*t)+…
                           x(5)*v0(5)*exp(-x(11)*v0(11)*t)+x(6)*v0(6)*exp(-x(12)*v0(12)*t));
   f=f+w(i)*(measured(i)-temp(i)-x(13)*v0(13)*dwell(i))^2.0;
end

%  ************** Compute Jacobian matrix jacob(i,np)  *************
%  For Modified LM, delta_x=inv([J b]'*[J b]+a*I)*[J b]'*delta_y.
%  Here, a is the regulator number,and J is the weighted Jacobian matrix
%  with sqrt(w(i))*jacob(i,j)=J(i,j).
%  delta_y(i)=sqrt(w(i))*(measured(i)-temp(i)). There are 13 unknown
%  parameters, including background count rate b.

for i=1:rn
   t=t0(i);
   w(i)=1.0/measured(i);
   for k=1:13
      if k<=6
         jacob(i,k)=count*dwell(i)*v0(k)*exp(-x(k+6)*v0(k+6)*t);
      elseif k<=12
          jacob(i,k)=(-t*v0(k))*x(k-6)*v0(k-6)*count*dwell(i)*exp(-x(k)*v0(k)*t);
      else
         jacob(i,k)=dwell(i)*v0(k);
      end
      J(i,k)=jacob(i,k)*sqrt(w(i));
   end
end
```

```
% *******  Define the function for pulse mode *********
function [f,J,temp]=LMfun13pulseRATIO(x)
count=4.0e5;
dt=0;
a10=0.033;a20=0.219;a30=0.196;a40=0.395;a50=0.115;a60=0.042;
r10=0.0124;r20=0.0305;r30=0.111;r40=0.301;r50=1.14;r60=3.01;
b=1.0;
va0=[a10; a20; a30; a40; a50; a60];
vr0=[r10; r20; r30; r40; r50; r60];
v0=[va0;vr0;b];
f=0.0;

%  Read Poisson data, which is generated from the corresponding seed in FORTRAN code.
[ti, keepin, measured, fit]=textread('pulse2.txt');

[rn]=size(measured);
dn=dt/0.025;

for i=1:rn
   if i<(401-dn)
      t0(i)=(i-0.5)*0.025+dt;
      dwell(i)=0.025;
   elseif i<(581-dn)
      t0(i)=10.0+(i-400-0.5+dn)*0.5;
      dwell(i)=0.5;
   else
      t0(i)=100.0+(i-580-0.5+dn)*5.0;
      dwell(i)=5.0;
   end
   t=t0(i);
   w(i)=1.0/measured(i);
   temp(i)=count*dwell(i)*(x(1)*v0(1)*x(7)*v0(7)*exp(-x(7)*v0(7)*t)+…
                           x(2)*v0(2)*x(8)*v0(8)*exp(-x(8)*v0(8)*t)+…
                           x(3)*v0(3)*x(9)*v0(9)*exp(-x(9)*v0(9)*t)+…
                           x(4)*v0(4)*x(10)*v0(10)*exp(-x(10)*v0(10)*t)+…
                           x(5)*v0(5)*x(11)*v0(11)*exp(-x(11)*v0(11)*t)+…
                           x(6)*v0(6)*x(12)*v0(12)*exp(-x(12)*v0(12)*t));
   f=f+w(i)*(measured(i)-temp(i)-x(13)*v0(13)*dwell(i))^2.0;
end

%  ************** Compute Jacobian matrix jacob(i, np)  *************
%  For Modified LM,delta_x=inv([J b]'*[J b]+a*I)*[J b]'*delta_y.
% Here, a is the regulator number,and J is the weighted Jacobian matrix
% with sqrt(w(i))*jacob(i,j)=J(i,j).
% delta_y(i)=sqrt(w(i))*(measured(i)-temp(i)). There are 13 unknown
% parameters, including background count rate b.
```

```
for i=1:rn
  t=t0(i);
  w(i)=1.0/measured(i);
  for k=1:13
    if k<=6
      jacob(i,k)=count*dwell(i)*v0(k)*x(k+6)*v0(k+6)*exp(-x(k+6)*v0(k+6)*t);
    elseif k<=12
       jacob(i,k)=count*dwell(i)*x(k-6)*v0(k-6)*v0(k)*exp(-x(k)*v0(k)*t)*…
                              (1.0-t*x(k)*v0(k));
    else
      jacob(i,k)=dwell(i)*v0(k);
    end
    J(i,k)=jacob(i,k)*sqrt(w(i));
  end
end


%   ************* Define function ridge  ****************
function [Xh, rss, Xh_ss, dof, HGout] = ridge(B, A, h)

%    For a linear model: B = A*X + noise
%    Given a regularization parameter h,
%    Just call: [Xh, rss, Xh_ss, dof] = ridge (B, A, h)
%    Xh is the vector of unknown parameters
%    If no regularization parameter h is given, just call:
%            [Xh, rss, Xh_ss, dof, h, G] = ridge(B, A);

 [n, p]= size(A);
 q = min(n, p);
 if nargin < 3
   hh = 1;
 else
   hh = length(h);
 end
 Xh = zeros (p, hh);
 rss = zeros (hh, 1);
 Xh_ss = zeros (hh, 1);
 dof = zeros (hh, 1);
 [U, S, V] = svd (A, 0);
 s = diag(S);
 s2 = s.^2;
 fc = U(:, 1:q)'*g;
 fs= s .* fc;
 if nargin < 3
   [h, G] = gcv(U, s, g);
   HGout(1) = {h};
   HGout(2) = {G};
 end
```

```
  for j = 1:hh
     Xh(:, j) = V(:, 1:q) * (fs./ (s2 + h(j)^2));
     Xh_ss(j) = sum(Xh(:, j).^2);
     rss(j) = h(j)^4 * sum(fc.^2 ./ (s2 + h(j)^2).^2);
     dof(j) = n - sum(s2./(s2 + h(j)^2));
  end
  if (n > p)
     rss = rss + sum((g - U(:, 1:q)*fc).^2);
  end

%  *********  Define function GCV ***************
% GCV  Generalized cross-validation.

function [rpar, G] = gcv(U, s, g)
  [n, q] = size(U);
  s = s(:);
  if length (s) ~= q
    error ('The value s should be the same vector of singular values.');
  end
  fc = U(:, 1:q)'*g;
  s2 = s.^2;
  rss0 = 0;
  if n > q
    rss0 = sum((g - U(:, 1:q)*fc).^2);
  end
    h_tol = ((q^2 + q + 1)*eps)^(1/2);
    h_max = max(s);
        h_min = min(s) * h_tol;

% Find the minimized value of GCV function
    minopt = optimset('TolX', h_tol, 'Display', 'off');
    [rpar, G] = fminbnd('gcvfunc', h_min, h_max, minopt, s2, fc, rss0, n-q);

% *********  Define function G  ********************
function G = gcvfunc (h, s2, fc, rss0, dof0)
  f = h^2 ./ (s2 + h^2);
  G = (norm (f.*fc)^2 + rss0) / (dof0 + sum(f))^2;
```

## A.5 Codes for L-curve and GCV Curve

```
% ***** This appendix contains the code used to plot L-curve and GCV curve.

B = [ ]; % !! Matrix B is obtained from the results in previous algorithms.
A = [ ]; % !!Matrix A is obtained from the results in previous algorithms.
% End of data reading.

[n, p]= size(A);
q = min(n, p);
noise_std = .001;
[U,s,V]= svd(X,0);
s = diag(s);
nh= 200;    % number of regularization parameters
h_max = s(1);
h_min= (eps)^(1/3);
h= h_max * (h_min/h_max).^( [0:nh-1] / (nh-1) );

[Xh, rss_h, Xh_ss, dof_h] = ridge (B, A, h);
% get regularization parameter from discrepancy principle
[d, idis] = min(abs( rss_h - n* noise_std^2 ));
h_dis = h(idis);
disp(['From discrepancy principle: h = ', num2str(h_dis)])

% plot the L-curve
h_l = 0.5;   % corner of L-curve
% get corresponding estimate of density variations
[Xhl, rss_hl, Xhl_ss] = ridge (B, A, h_l);
ihs = zeros(q, 1);
for k = 1:q
  [dum, i] = min(abs( s(k)-h ));
  ihs(k) = i;
end
figure(1)
clf
loglog(rss_h, Xh_ss, 'k-', ...
      rss_h(ihs), Xh_ss(ihs), 'kx')
hold on
% label the singular values
for k = 1:q
  hd = text(rss_h(ihs(k)), Xh_ss(ihs(k)), num2str(h(ihs(k))));
  set(hd, 'VerticalAlignment', 'bottom')
end
% mark corner of L-curve
loglog([1e-3 rss_hl], [Xhl_ss Xhl_ss], 'k--', ...
      [rss_hl rss_hl], [1 Xhl_ss], 'k--')
set(gca, 'Box', 'on')
axis([1e2 1e3 1e-12 1e3])
```

```
xlabel('Residual SS (||AX-b||**2)')
ylabel('Estimate SS (||X||**2)')
disp(['From L-curve: h = ', num2str(h_l)]);

% plot GCV
G = rss_h ./ dof_h.^2;
[f_hgcv, rss_hgcv, f_hgcv_ss, dof_hgcv, h_gcv, G_hgcv] = ...
   ridge(B, A);
disp(['From GCV: h = ', num2str(h_gcv)]);
figure(2)
clf
loglog(h, G, 'k-', ...
      [h_gcv h_gcv], [10^(-3) G_hgcv], 'k--')
set(gca, 'Box', 'on')
xlabel('Regularization parameter h')
ylabel('GCV function')
axis([1e-5 1e6 2e-4 1e-2])
```

# APPENDIX B

# VALIDATION OF PROGRAM CODES

To verify the validation of the program codes (Matrix Inverse with SVD, Levenberg-Marquardt, quasi-Newton, Ridge Regression), a nonlinear least squares equation is listed as an example to check the calculated results from above numerical methods. The function is:

$$F = 2(x_1 - 2x_2)^2 + (1 - x_1)^2$$

Set the partial differentiation of function F with respect to $x_1$ and $x_2$ to zero for optimization. The new equation can be expressed in matrix form as:

$$\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 3 & -4 \\ 1 & -2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

The above matrix will be used in Matrix Inverse and Ridge Regression methods. The codes are listed in Appendices B.1, B.2, B.3 (FORTRAN), and B.4 (MATLAB).

Set the initial guessed values as: $x_1$=2.0, $x_2$=2.0, the calculated results are:

| Algorithm | $x_1$ | $x_2$ | F |
|---|---|---|---|
| MI-SVD | 1.0 | 0.5 | 0 |
| LM | 1.0 | 0.5 | 0 |
| QN | 1.0 | 0.5 | 0 |
| RR | 1.0 | 0.5 | 5.92E-31 |

Note: The F value from RR algorithm is not zero. This is due to the machine precision. It is almost equal to zero.

## B.1 Matrix Inverse Algorithm

```
USE NUMBERICAL-LIBRARIES
     INTEGER I,J,L,K,M,NN,NM,COUNT
          INTEGER IPATH,LDA,NRA,NCA,LDU,LDV
          REAL*8 X0(2),X(2),H(2,2),F,TOL
          REAL*8 UH(2,2),VH(2,2),SH(2),D(2)
          REAL*8 SHMIN, SHMAX, CVM_H(2,2)
          PARAMETER (NM=2,NN=2,NT=2, IPATH=11)
         PARAMETER (NRA=2,NCA=2,LDA=NRA,LDU=NRA,LDV=NCA)

          TOL=10.*AMACH(6)
          H(1,1)=3.0
          H(1,2)=-4.0
          H(2,1)=1.0
          H(2,2)=-2.0
          D(1)=1.0
          D(2)=0.0
C          ****************** Matrix Inverse **************************
          CALL
          DLSVRR(NRA,NCA,H,LDA,IPATH,TOL,IRANK,SH,UH,LDU,VH,LDV)
          SHMAX=0.
          DO J=1,2
                    IF(SH(J).GT.SHMAX) SHMAX=SH(J)
          ENDDO
          SHMIN=SHMAX*1.0E-8
          DO J=1,2
                    IF(SH(J).LT.SHMIN) SH(J)=0.
          ENDDO
          CALL SVBKSB(UH,SH,VH,NM,NN,NM,NN,D,X)
          CALL SVDVAR(VH,NN,NN,SH,CVM_H,NN)
          F=2.0*(X(1)-2.0*X(2))**2+(1.0-X(1))**2

          WRITE (*,901) (X(J), J=1,2)
          WRITE (*,903) F
901       FORMAT("THE SOLUTION IS:  ", //,2(F12.5,/))
903       FORMAT (" FUNCTION VALUE F=",F12.5)
          END
C          ********** SUBROUTINE SVDVAR ******************
          SUBROUTINE SVDVAR(V,MA,NP,WT,CVM,NCVM)
          PARAMETER (MMAX=20)
          REAL*8 V(NP,NP),WT(NP),CVM(NCVM,NCVM),WTI(MMAX)
          DO I=1,MA
                    WTI(I)=0.0
```

```
              IF(WT(I).NE.0.0) WTI(I)=1.0/(WT(I)*WT(I))
      ENDDO
      DO I=1,MA
              DO J=1,I
                      SUM=0.0
                      DO K=1,MA
                              SUM=SUM+V(I,K)*V(J,K)*WTI(K)
                      ENDDO
                      CVM(I,J)=SUM
                      CVM(J,I)=SUM
              ENDDO
      ENDDO
      RETURN
      END

C      ******************** Subroutine SVBKSB ********************
C      Solves A.X=B for a vector X. Where X=V.[diag(1/wj)].(U'.B)

      SUBROUTINE SVBKSB(U,WT,V,M,N,MP,NP,B,X)
      PARAMETER (NMAX=20)
      REAL*8 U(MP,NP),WT(NP),V(NP,NP),B(MP),X(NP),TMP(NMAX)
      DO J=1,N
      S=0.
              IF(WT(J).NE.0.) THEN
                      DO I=1,M
                              S=S+U(I,J)*B(I)
                      ENDDO
                      S=S/WT(J)
              ENDIF
      TMP(J)=S
      ENDDO
      DO J=1,N
              S=0.
              DO JJ=1,N
                      S=S+V(J,JJ)*TMP(JJ)
              ENDDO
              X(J)=S
      ENDDO
      RETURN
      END
```

## B.2 Levenberg-Marquardt Algorithm

```
        USE NUMERICAL_LIBRARIES
        INTEGER N,I,K,M,LDFJAC,NT
        PARAMETER (LDFJAC=2,M=2,N=2,NT=2)
        INTEGER IPARAM(7),L,NOUT,F
        REAL*8 RPARAM(7), X(N),XSCALE(N),TF(NT),XGUESS(N)
        EXTERNAL FCN
        REAL*8 FVEC(M),FJAC(M,N),FSCALE(M)
C       ******** SET INITIAL GUESS VALUES  **********************
        DO I=1,2
                XGUESS(I)=2.0
                XSCALE(I)=0.5
        ENDDO
        IPARAM(1) = 1
        IPARAM(3) = 2000
        IPARAM(4) = 2000
        IPARAM(5) = 2000
        IPARAM(6) = 1
        FSCALE =0.005
        CALL DU4LSF (IPARAM,RPARAM)
        RPARAM(4)=2.0*RPARAM(4)
        CALL DUNLSF (FCN,M,N,XGUESS,XSCALE,FSCALE,IPARAM,
    & RPARAM,X,FVEC,FJAC,LDFJAC)
        CALL UMACH (2,NOUT)
        F=2.0*(X(1)-2.0*X(2))**2+(1.0-X(1))**2

        WRITE (6,901) (X(L), L=1,2)
901     FORMAT("THE SOLUTION IS ", //,2(F12.5,/))
        WRITE (6,902) F
902     FORMAT ("THE FUNCTION VALUE F:",F12.5)
        END

C       ********************DEFINE FUNCTION*********************
        SUBROUTINE FCN(J,N,X,TF)
        INTEGER N,I,J
        PARAMETER (NT=2)
        REAL*8 VALUES(2), X(2),TF(NT)
        TF(1)=SQRT(2.0)*(X(1)-2.0*X(2))
        TF(2)=1-X(1)
        RETURN
        END
```

## B.3 Quasi-Newton Algorithm

```
USE NUMERICAL_LIBRARIES
        INTEGER N,I,NT
        PARAMETER (N=2,NT=2)
        INTEGER IPARAM(7),L, NOUT
        REAL*8 F,FSCALE, RPARAM(7),X(2),XSCALE(2),Y(2)
        EXTERNAL DIFFSQUARES

        DO I=1,2
          Y(I) = 2.0
          XSCALE(I) =50.0
        ENDDO
        IPARAM(1) = 1
        IPARAM(3) = 2000
        IPARAM(4) = 2000
        IPARAM(5) = 2000
        IPARAM(6) = 1
        FSCALE =0.5

        CALL DIFFSQUARES(N,Y,F)
        CALL DUMINF(DIFFSQUARES,N,Y, XSCALE, FSCALE,IPARAM,
RPARAM,X,F)
        RPARAM(1)=2.0*RPARAM(1)
        CALL UMACH (2, NOUT)
        WRITE (6,901) (X(L), L=1,2)
901     FORMAT("THE SOLUTION IS ", //,2(F12.5,/))
        WRITE (6,902) F
902     FORMAT ("THE FUNCTION VALUE F:",F12.5)
        F=2.0*(X(1)-2.0*X(2))**2+(1.0-X(1))**2
        END

C       ******************* DEFINE FUNCTION *******************
        SUBROUTINE DIFFSQUARES(N,X,F)
        INTEGER N
        REAL*8 X(2),F

        F=2.0*(X(1)-2.0*X(2))**2+(1.0-X(1))**2
        RETURN
        END
```

## B.4 Ridge Regression Algorithm

```
clear all;
format short e
% *** Define the initial guessed values ********
for i=1:2
    x0(i)=2.0;
end
x=x0';
[f,J]=LMfun13noiseRATIO(x);
[f,J]=LMfun13noiseRATIO(x);
    for i=1:2
        delta_y(1,1)=1.0;
        delta_y(2,1)=0.0;
    end
%  regularization method  **************************
 %[delta_x,rss,delta_x_ss,dof,h,G]=ridge(delta_y,J);
h=0.0;
[delta_x,rss,delta_x_ss,dof]=ridge(delta_y,J,h);
for i=1:2
    x(i)=delta_x(i);
end
%  ********* end of Ridge Regression ****************
    [f,J]=LMfun13noiseRATIO(x);

function [f,J]=LMfun13noiseRATIO(x)

f=2.0*(x(1)-2.0*x(2))^2+(1.0-x(1))^2;
%  ********* Jacobian matrix J(i,np)  *********
J(1,1)=3.0;
J(1,2)=-4.0;
J(2,1)=1.0;
J(2,2)=-2.0;
```

Note: The function “ridge” is the same as that in Appendix A.4

# APPENDIX C

# EXPERIMENTAL DATA

| Saturation Mode | | | | | | Pulse Mode | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Time | Counts | Time | Counts | Time | Counts | Time | Counts | Time | Counts | Time | Counts |
| 0.468 | 997 | 1.393 | 825 | 2.318 | 715 | 0.475 | 1133 | 1.4 | 760 | 2.325 | 555 |
| 0.493 | 1001 | 1.418 | 808 | 2.343 | 660 | 0.5 | 1083 | 1.425 | 749 | 2.35 | 586 |
| 0.518 | 973 | 1.443 | 808 | 2.368 | 693 | 0.525 | 1092 | 1.45 | 783 | 2.375 | 552 |
| 0.543 | 982 | 1.468 | 776 | 2.393 | 692 | 0.55 | 1115 | 1.475 | 771 | 2.4 | 557 |
| 0.568 | 925 | 1.493 | 800 | 2.418 | 682 | 0.575 | 1107 | 1.5 | 748 | 2.425 | 570 |
| 0.593 | 986 | 1.518 | 775 | 2.443 | 687 | 0.6 | 1103 | 1.525 | 699 | 2.45 | 517 |
| 0.618 | 955 | 1.543 | 812 | 2.468 | 704 | 0.625 | 1075 | 1.55 | 727 | 2.475 | 533 |
| 0.643 | 904 | 1.568 | 770 | 2.493 | 661 | 0.65 | 1061 | 1.575 | 674 | 2.5 | 513 |
| 0.668 | 975 | 1.593 | 786 | 2.518 | 701 | 0.675 | 1064 | 1.6 | 700 | 2.525 | 509 |
| 0.693 | 960 | 1.618 | 799 | 2.543 | 703 | 0.7 | 1031 | 1.625 | 682 | 2.55 | 493 |
| 0.718 | 924 | 1.643 | 735 | 2.568 | 645 | 0.725 | 1041 | 1.65 | 733 | 2.575 | 513 |
| 0.743 | 936 | 1.668 | 739 | 2.593 | 661 | 0.75 | 978 | 1.675 | 680 | 2.6 | 548 |
| 0.768 | 909 | 1.693 | 762 | 2.618 | 653 | 0.775 | 965 | 1.7 | 691 | 2.625 | 526 |
| 0.793 | 916 | 1.718 | 760 | 2.643 | 709 | 0.8 | 977 | 1.725 | 726 | 2.65 | 511 |
| 0.818 | 934 | 1.743 | 761 | 2.668 | 658 | 0.825 | 957 | 1.75 | 630 | 2.675 | 505 |
| 0.843 | 950 | 1.768 | 750 | 2.693 | 690 | 0.85 | 1001 | 1.775 | 681 | 2.7 | 517 |
| 0.868 | 951 | 1.793 | 746 | 2.718 | 653 | 0.875 | 966 | 1.8 | 629 | 2.725 | 481 |
| 0.893 | 877 | 1.818 | 782 | 2.743 | 711 | 0.9 | 915 | 1.825 | 634 | 2.75 | 468 |
| 0.918 | 935 | 1.843 | 741 | 2.768 | 661 | 0.925 | 899 | 1.85 | 614 | 2.775 | 481 |
| 0.943 | 892 | 1.868 | 749 | 2.793 | 657 | 0.95 | 872 | 1.875 | 666 | 2.8 | 470 |
| 0.968 | 962 | 1.893 | 745 | 2.818 | 639 | 0.975 | 865 | 1.9 | 616 | 2.825 | 481 |
| 0.993 | 959 | 1.918 | 725 | 2.843 | 671 | 1 | 854 | 1.925 | 641 | 2.85 | 504 |
| 1.018 | 887 | 1.943 | 768 | 2.868 | 672 | 1.025 | 866 | 1.95 | 629 | 2.875 | 503 |
| 1.043 | 899 | 1.968 | 733 | 2.893 | 642 | 1.05 | 813 | 1.975 | 610 | 2.9 | 466 |
| 1.068 | 917 | 1.993 | 685 | 2.918 | 631 | 1.075 | 833 | 2 | 626 | 2.925 | 442 |
| 1.093 | 804 | 2.018 | 721 | 2.943 | 642 | 1.1 | 877 | 2.025 | 590 | 2.95 | 498 |
| 1.118 | 924 | 2.043 | 732 | 2.968 | 654 | 1.125 | 855 | 2.05 | 628 | 2.975 | 474 |
| 1.143 | 846 | 2.068 | 718 | 2.993 | 619 | 1.15 | 784 | 2.075 | 574 | 3 | 454 |
| 1.168 | 855 | 2.093 | 735 | 3.018 | 636 | 1.175 | 779 | 2.1 | 581 | 3.025 | 435 |
| 1.193 | 854 | 2.118 | 743 | 3.043 | 658 | 1.2 | 797 | 2.125 | 594 | 3.05 | 461 |
| 1.218 | 854 | 2.143 | 701 | 3.068 | 606 | 1.225 | 783 | 2.15 | 584 | 3.075 | 462 |
| 1.243 | 858 | 2.168 | 742 | 3.093 | 654 | 1.25 | 815 | 2.175 | 572 | 3.1 | 448 |
| 1.268 | 840 | 2.193 | 704 | 3.118 | 627 | 1.275 | 783 | 2.2 | 594 | 3.125 | 442 |
| 1.293 | 828 | 2.218 | 701 | 3.143 | 600 | 1.3 | 766 | 2.225 | 582 | 3.15 | 450 |
| 1.318 | 765 | 2.243 | 734 | 3.168 | 634 | 1.325 | 759 | 2.25 | 543 | 3.175 | 436 |
| 1.343 | 798 | 2.268 | 735 | 3.193 | 624 | 1.35 | 761 | 2.275 | 560 | 3.2 | 452 |
| 1.368 | 807 | 2.293 | 725 | 3.218 | 618 | 1.375 | 742 | 2.3 | 537 | 3.225 | 433 |

| Saturation Mode | | | | | | Pulse Mode | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Time | Counts | Time | Counts | Time | Counts | Time | Counts | Time | Counts | Time | Counts |
| 3.243 | 627 | 4.193 | 577 | 5.143 | 523 | 3.25 | 443 | 4.2 | 319 | 5.15 | 268 |
| 3.268 | 611 | 4.218 | 543 | 5.168 | 464 | 3.275 | 443 | 4.225 | 372 | 5.175 | 270 |
| 3.293 | 634 | 4.243 | 589 | 5.193 | 458 | 3.3 | 418 | 4.25 | 329 | 5.2 | 293 |
| 3.318 | 624 | 4.268 | 566 | 5.218 | 502 | 3.325 | 449 | 4.275 | 350 | 5.225 | 287 |
| 3.343 | 647 | 4.293 | 553 | 5.243 | 452 | 3.35 | 439 | 4.3 | 341 | 5.25 | 280 |
| 3.368 | 613 | 4.318 | 545 | 5.268 | 469 | 3.375 | 386 | 4.325 | 330 | 5.275 | 285 |
| 3.393 | 615 | 4.343 | 532 | 5.293 | 518 | 3.4 | 407 | 4.35 | 362 | 5.3 | 277 |
| 3.418 | 611 | 4.368 | 519 | 5.318 | 507 | 3.425 | 441 | 4.375 | 324 | 5.325 | 262 |
| 3.443 | 601 | 4.393 | 533 | 5.343 | 510 | 3.45 | 416 | 4.4 | 332 | 5.35 | 261 |
| 3.468 | 619 | 4.418 | 567 | 5.368 | 500 | 3.475 | 392 | 4.425 | 301 | 5.375 | 261 |
| 3.493 | 565 | 4.443 | 516 | 5.393 | 486 | 3.5 | 400 | 4.45 | 332 | 5.4 | 250 |
| 3.518 | 623 | 4.468 | 548 | 5.418 | 468 | 3.525 | 416 | 4.475 | 352 | 5.425 | 260 |
| 3.543 | 579 | 4.493 | 537 | 5.443 | 512 | 3.55 | 411 | 4.5 | 340 | 5.45 | 268 |
| 3.568 | 580 | 4.518 | 534 | 5.468 | 501 | 3.575 | 401 | 4.525 | 300 | 5.475 | 267 |
| 3.593 | 585 | 4.543 | 516 | 5.493 | 451 | 3.6 | 398 | 4.55 | 320 | 5.5 | 267 |
| 3.618 | 589 | 4.568 | 478 | 5.518 | 484 | 3.625 | 411 | 4.575 | 315 | 5.525 | 282 |
| 3.643 | 595 | 4.593 | 538 | 5.543 | 479 | 3.65 | 420 | 4.6 | 307 | 5.55 | 266 |
| 3.668 | 586 | 4.618 | 507 | 5.568 | 469 | 3.675 | 385 | 4.625 | 315 | 5.575 | 253 |
| 3.693 | 617 | 4.643 | 538 | 5.593 | 439 | 3.7 | 380 | 4.65 | 284 | 5.6 | 246 |
| 3.718 | 583 | 4.668 | 556 | 5.618 | 497 | 3.725 | 344 | 4.675 | 329 | 5.625 | 245 |
| 3.743 | 570 | 4.693 | 507 | 5.643 | 458 | 3.75 | 373 | 4.7 | 323 | 5.65 | 256 |
| 3.768 | 560 | 4.718 | 526 | 5.668 | 479 | 3.775 | 374 | 4.725 | 298 | 5.675 | 257 |
| 3.793 | 552 | 4.743 | 499 | 5.693 | 469 | 3.8 | 385 | 4.75 | 328 | 5.7 | 236 |
| 3.818 | 565 | 4.768 | 469 | 5.718 | 462 | 3.825 | 392 | 4.775 | 300 | 5.725 | 266 |
| 3.843 | 574 | 4.793 | 513 | 5.743 | 465 | 3.85 | 397 | 4.8 | 314 | 5.75 | 248 |
| 3.868 | 558 | 4.818 | 506 | 5.768 | 447 | 3.875 | 370 | 4.825 | 318 | 5.775 | 244 |
| 3.893 | 550 | 4.843 | 525 | 5.793 | 433 | 3.9 | 357 | 4.85 | 327 | 5.8 | 261 |
| 3.918 | 557 | 4.868 | 543 | 5.818 | 475 | 3.925 | 393 | 4.875 | 289 | 5.825 | 242 |
| 3.943 | 565 | 4.893 | 509 | 5.843 | 491 | 3.95 | 347 | 4.9 | 286 | 5.85 | 241 |
| 3.968 | 568 | 4.918 | 464 | 5.868 | 438 | 3.975 | 354 | 4.925 | 316 | 5.875 | 265 |
| 3.993 | 576 | 4.943 | 501 | 5.893 | 457 | 4 | 369 | 4.95 | 285 | 5.9 | 226 |
| 4.018 | 570 | 4.968 | 501 | 5.918 | 465 | 4.025 | 347 | 4.975 | 272 | 5.925 | 255 |
| 4.043 | 595 | 4.993 | 497 | 5.943 | 481 | 4.05 | 383 | 5 | 317 | 5.95 | 242 |
| 4.068 | 539 | 5.018 | 496 | 5.968 | 496 | 4.075 | 356 | 5.025 | 274 | 5.975 | 220 |
| 4.093 | 534 | 5.043 | 492 | 5.993 | 444 | 4.1 | 346 | 5.05 | 297 | 6 | 234 |
| 4.118 | 597 | 5.068 | 497 | 6.018 | 445 | 4.125 | 350 | 5.075 | 268 | 6.025 | 230 |
| 4.143 | 563 | 5.093 | 540 | 6.043 | 414 | 4.15 | 347 | 5.1 | 285 | 6.05 | 224 |
| 4.168 | 548 | 5.118 | 499 | 6.068 | 429 | 4.175 | 346 | 5.125 | 302 | 6.075 | 236 |

| Saturation Mode | | | | | | Pulse Mode | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Time | Counts | Time | Counts | Time | Counts | Time | Counts | Time | Counts | Time | Counts |
| 6.093 | 468 | 7.043 | 410 | 7.993 | 400 | 6.1 | 226 | 7.05 | 215 | 8 | 166 |
| 6.118 | 447 | 7.068 | 433 | 8.018 | 381 | 6.125 | 233 | 7.075 | 189 | 8.025 | 140 |
| 6.143 | 401 | 7.093 | 424 | 8.043 | 373 | 6.15 | 228 | 7.1 | 179 | 8.05 | 166 |
| 6.168 | 434 | 7.118 | 393 | 8.068 | 382 | 6.175 | 247 | 7.125 | 199 | 8.075 | 161 |
| 6.193 | 416 | 7.143 | 429 | 8.093 | 371 | 6.2 | 231 | 7.15 | 191 | 8.1 | 143 |
| 6.218 | 436 | 7.168 | 365 | 8.118 | 393 | 6.225 | 225 | 7.175 | 176 | 8.125 | 158 |
| 6.243 | 494 | 7.193 | 422 | 8.143 | 366 | 6.25 | 200 | 7.2 | 206 | 8.15 | 161 |
| 6.268 | 469 | 7.218 | 439 | 8.168 | 368 | 6.275 | 231 | 7.225 | 181 | 8.175 | 157 |
| 6.293 | 437 | 7.243 | 431 | 8.193 | 400 | 6.3 | 204 | 7.25 | 182 | 8.2 | 182 |
| 6.318 | 446 | 7.268 | 383 | 8.218 | 382 | 6.325 | 229 | 7.275 | 205 | 8.225 | 165 |
| 6.343 | 454 | 7.293 | 386 | 8.243 | 372 | 6.35 | 215 | 7.3 | 178 | 8.25 | 177 |
| 6.368 | 453 | 7.318 | 411 | 8.268 | 384 | 6.375 | 218 | 7.325 | 179 | 8.275 | 170 |
| 6.393 | 461 | 7.343 | 396 | 8.293 | 392 | 6.4 | 225 | 7.35 | 187 | 8.3 | 162 |
| 6.418 | 446 | 7.368 | 397 | 8.318 | 337 | 6.425 | 214 | 7.375 | 178 | 8.325 | 160 |
| 6.443 | 422 | 7.393 | 432 | 8.343 | 385 | 6.45 | 217 | 7.4 | 188 | 8.35 | 173 |
| 6.468 | 411 | 7.418 | 415 | 8.368 | 364 | 6.475 | 203 | 7.425 | 163 | 8.375 | 139 |
| 6.493 | 430 | 7.443 | 415 | 8.393 | 376 | 6.5 | 231 | 7.45 | 188 | 8.4 | 151 |
| 6.518 | 449 | 7.468 | 406 | 8.418 | 346 | 6.525 | 206 | 7.475 | 204 | 8.425 | 147 |
| 6.543 | 425 | 7.493 | 393 | 8.443 | 387 | 6.55 | 209 | 7.5 | 168 | 8.45 | 166 |
| 6.568 | 405 | 7.518 | 377 | 8.468 | 388 | 6.575 | 221 | 7.525 | 205 | 8.475 | 148 |
| 6.593 | 390 | 7.543 | 404 | 8.493 | 386 | 6.6 | 229 | 7.55 | 161 | 8.5 | 154 |
| 6.618 | 419 | 7.568 | 374 | 8.518 | 375 | 6.625 | 216 | 7.575 | 159 | 8.525 | 166 |
| 6.643 | 400 | 7.593 | 398 | 8.543 | 354 | 6.65 | 216 | 7.6 | 189 | 8.55 | 143 |
| 6.668 | 440 | 7.618 | 378 | 8.568 | 334 | 6.675 | 206 | 7.625 | 184 | 8.575 | 140 |
| 6.693 | 429 | 7.643 | 382 | 8.593 | 353 | 6.7 | 226 | 7.65 | 192 | 8.6 | 146 |
| 6.718 | 405 | 7.668 | 353 | 8.618 | 377 | 6.725 | 206 | 7.675 | 182 | 8.625 | 170 |
| 6.743 | 440 | 7.693 | 402 | 8.643 | 357 | 6.75 | 199 | 7.7 | 189 | 8.65 | 135 |
| 6.768 | 420 | 7.718 | 402 | 8.668 | 358 | 6.775 | 211 | 7.725 | 161 | 8.675 | 136 |
| 6.793 | 441 | 7.743 | 410 | 8.693 | 361 | 6.8 | 219 | 7.75 | 177 | 8.7 | 155 |
| 6.818 | 396 | 7.768 | 361 | 8.718 | 328 | 6.825 | 227 | 7.775 | 173 | 8.725 | 143 |
| 6.843 | 384 | 7.793 | 383 | 8.743 | 339 | 6.85 | 207 | 7.8 | 186 | 8.75 | 144 |
| 6.868 | 398 | 7.818 | 355 | 8.768 | 347 | 6.875 | 198 | 7.825 | 171 | 8.775 | 148 |
| 6.893 | 425 | 7.843 | 426 | 8.793 | 354 | 6.9 | 223 | 7.85 | 175 | 8.8 | 114 |
| 6.918 | 420 | 7.868 | 400 | 8.818 | 367 | 6.925 | 208 | 7.875 | 151 | 8.825 | 140 |
| 6.943 | 390 | 7.893 | 354 | 8.843 | 386 | 6.95 | 200 | 7.9 | 173 | 8.85 | 154 |
| 6.968 | 391 | 7.918 | 376 | 8.868 | 350 | 6.975 | 202 | 7.925 | 178 | 8.875 | 151 |
| 6.993 | 424 | 7.943 | 399 | 8.893 | 359 | 7 | 207 | 7.95 | 172 | 8.9 | 128 |
| 7.018 | 409 | 7.968 | 410 | 8.918 | 352 | 7.025 | 216 | 7.975 | 150 | 8.925 | 130 |

| Saturation Mode | | | | | | Pulse Mode | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Time | Counts | Time | Counts | Time | Counts | Time | Counts | Time | Counts | Time | Counts |
| 8.943 | 346 | 9.893 | 346 | 18.443 | 4302 | 8.95 | 127 | 9.9 | 125 | 18.45 | 1060 |
| 8.968 | 360 | 9.918 | 390 | 18.943 | 4182 | 8.975 | 149 | 9.925 | 112 | 18.95 | 1004 |
| 8.993 | 366 | 9.943 | 340 | 19.443 | 4168 | 9 | 132 | 9.95 | 115 | 19.45 | 913 |
| 9.018 | 357 | 9.968 | 328 | 19.943 | 4089 | 9.025 | 136 | 9.975 | 131 | 19.95 | 907 |
| 9.043 | 347 | 9.993 | 342 | 20.443 | 3904 | 9.05 | 128 | 10 | 113 | 20.45 | 865 |
| 9.068 | 357 | 10.018 | 338 | 20.943 | 3978 | 9.075 | 155 | 10.025 | 115 | 20.95 | 811 |
| 9.093 | 303 | 10.043 | 332 | 21.443 | 3871 | 9.1 | 144 | 10.05 | 110 | 21.45 | 865 |
| 9.118 | 375 | 10.068 | 334 | 21.943 | 3898 | 9.125 | 160 | 10.075 | 125 | 21.95 | 826 |
| 9.143 | 355 | 10.093 | 316 | 22.443 | 3754 | 9.15 | 132 | 10.1 | 126 | 22.45 | 827 |
| 9.168 | 375 | 10.118 | 330 | 22.943 | 3612 | 9.175 | 121 | 10.125 | 116 | 22.95 | 735 |
| 9.193 | 370 | 10.143 | 326 | 23.443 | 3561 | 9.2 | 126 | 10.15 | 146 | 23.45 | 777 |
| 9.218 | 337 | 10.168 | 315 | 23.943 | 3495 | 9.225 | 128 | 10.175 | 130 | 23.95 | 732 |
| 9.243 | 357 | 10.193 | 345 | 24.443 | 3455 | 9.25 | 129 | 10.2 | 109 | 24.45 | 749 |
| 9.268 | 352 | 10.218 | 327 | 24.943 | 3380 | 9.275 | 138 | 10.225 | 111 | 24.95 | 699 |
| 9.293 | 358 | 10.243 | 314 | 25.443 | 3282 | 9.3 | 141 | 10.25 | 112 | 25.45 | 643 |
| 9.318 | 354 | 10.268 | 306 | 25.943 | 3266 | 9.325 | 143 | 10.275 | 116 | 25.95 | 677 |
| 9.343 | 343 | 10.293 | 331 | 26.443 | 3187 | 9.35 | 126 | 10.3 | 118 | 26.45 | 623 |
| 9.368 | 340 | 10.318 | 314 | 26.943 | 3214 | 9.375 | 135 | 10.325 | 113 | 26.95 | 586 |
| 9.393 | 353 | 10.343 | 305 | 27.443 | 3081 | 9.4 | 131 | 10.35 | 113 | 27.45 | 610 |
| 9.418 | 334 | 10.368 | 324 | 27.943 | 3017 | 9.425 | 130 | 10.375 | 104 | 27.95 | 590 |
| 9.443 | 355 | 10.393 | 360 | 28.443 | 2981 | 9.45 | 131 | 10.4 | 116 | 28.45 | 568 |
| 9.468 | 330 | 10.418 | 308 | 28.943 | 2906 | 9.475 | 138 | 10.425 | 114 | 28.95 | 570 |
| 9.493 | 344 | 10.443 | 300 | 29.443 | 2856 | 9.5 | 138 | 10.45 | 119 | 29.45 | 534 |
| 9.518 | 372 | 10.943 | 6358 | 29.943 | 2808 | 9.525 | 138 | 10.95 | 2206 | 29.95 | 520 |
| 9.543 | 345 | 11.443 | 6073 | 30.443 | 2862 | 9.55 | 119 | 11.45 | 2077 | 30.45 | 477 |
| 9.568 | 335 | 11.943 | 5985 | 30.943 | 2814 | 9.575 | 120 | 11.95 | 1940 | 30.95 | 506 |
| 9.593 | 328 | 12.443 | 5828 | 31.443 | 2658 | 9.6 | 127 | 12.45 | 1810 | 31.45 | 492 |
| 9.618 | 354 | 12.943 | 5636 | 31.943 | 2660 | 9.625 | 128 | 12.95 | 1680 | 31.95 | 466 |
| 9.643 | 312 | 13.443 | 5480 | 32.443 | 2712 | 9.65 | 128 | 13.45 | 1557 | 32.45 | 461 |
| 9.668 | 340 | 13.943 | 5430 | 32.943 | 2642 | 9.675 | 131 | 13.95 | 1477 | 32.95 | 479 |
| 9.693 | 333 | 14.443 | 5104 | 33.443 | 2541 | 9.7 | 116 | 14.45 | 1510 | 33.45 | 437 |
| 9.718 | 344 | 14.943 | 5017 | 33.943 | 2512 | 9.725 | 123 | 14.95 | 1410 | 33.95 | 418 |
| 9.743 | 309 | 15.443 | 4978 | 34.443 | 2452 | 9.75 | 131 | 15.45 | 1381 | 34.45 | 412 |
| 9.768 | 327 | 15.943 | 4822 | 34.943 | 2461 | 9.775 | 128 | 15.95 | 1302 | 34.95 | 471 |
| 9.793 | 343 | 16.443 | 4680 | 35.443 | 2375 | 9.8 | 132 | 16.45 | 1238 | 35.45 | 420 |
| 9.818 | 344 | 16.943 | 4513 | 35.943 | 2333 | 9.825 | 112 | 16.95 | 1244 | 35.95 | 441 |
| 9.843 | 337 | 17.443 | 4564 | 36.443 | 2357 | 9.85 | 136 | 17.45 | 1120 | 36.45 | 416 |
| 9.868 | 331 | 17.943 | 4278 | 36.943 | 2343 | 9.875 | 119 | 17.95 | 1015 | 36.95 | 383 |

| Saturation Mode | | | | | | Pulse Mode | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Time | Counts | Time | Counts | Time | Counts | Time | Counts | Time | Counts | Time | Counts |
| 37.443 | 2323 | 56.443 | 1392 | 75.443 | 791 | 37.45 | 408 | 56.45 | 205 | 75.45 | 115 |
| 37.943 | 2222 | 56.943 | 1392 | 75.943 | 860 | 37.95 | 377 | 56.95 | 218 | 75.95 | 117 |
| 38.443 | 2181 | 57.443 | 1263 | 76.443 | 846 | 38.45 | 336 | 57.45 | 214 | 76.45 | 94 |
| 38.943 | 2220 | 57.943 | 1260 | 76.943 | 782 | 38.95 | 365 | 57.95 | 188 | 76.95 | 121 |
| 39.443 | 2074 | 58.443 | 1281 | 77.443 | 763 | 39.45 | 324 | 58.45 | 224 | 77.45 | 112 |
| 39.943 | 2087 | 58.943 | 1199 | 77.943 | 801 | 39.95 | 311 | 58.95 | 187 | 77.95 | 103 |
| 40.443 | 2120 | 59.443 | 1252 | 78.443 | 786 | 40.45 | 321 | 59.45 | 199 | 78.45 | 79 |
| 40.943 | 2050 | 59.943 | 1179 | 78.943 | 771 | 40.95 | 318 | 59.95 | 182 | 78.95 | 115 |
| 41.443 | 1992 | 60.443 | 1183 | 79.443 | 744 | 41.45 | 333 | 60.45 | 182 | 79.45 | 173 |
| 41.943 | 2004 | 60.943 | 1217 | 79.943 | 764 | 41.95 | 368 | 60.95 | 171 | 79.95 | 101 |
| 42.443 | 1933 | 61.443 | 1140 | 80.443 | 721 | 42.45 | 350 | 61.45 | 162 | 80.45 | 97 |
| 42.943 | 1967 | 61.943 | 1247 | 80.943 | 743 | 42.95 | 352 | 61.95 | 171 | 80.95 | 98 |
| 43.443 | 1910 | 62.443 | 1206 | 81.443 | 752 | 43.45 | 323 | 62.45 | 163 | 81.45 | 93 |
| 43.943 | 1918 | 62.943 | 1166 | 81.943 | 737 | 43.95 | 327 | 62.95 | 191 | 81.95 | 98 |
| 44.443 | 1869 | 63.443 | 1101 | 82.443 | 720 | 44.45 | 314 | 63.45 | 185 | 82.45 | 84 |
| 44.943 | 1824 | 63.943 | 1087 | 82.943 | 783 | 44.95 | 280 | 63.95 | 150 | 82.95 | 91 |
| 45.443 | 1831 | 64.443 | 1061 | 83.443 | 711 | 45.45 | 304 | 64.45 | 191 | 83.45 | 95 |
| 45.943 | 1743 | 64.943 | 1105 | 83.943 | 707 | 45.95 | 260 | 64.95 | 166 | 83.95 | 87 |
| 46.443 | 1668 | 65.443 | 1064 | 84.443 | 644 | 46.45 | 292 | 65.45 | 147 | 84.45 | 94 |
| 46.943 | 1726 | 65.943 | 1009 | 84.943 | 658 | 46.95 | 283 | 65.95 | 140 | 84.95 | 78 |
| 47.443 | 1744 | 66.443 | 988 | 85.443 | 628 | 47.45 | 306 | 66.45 | 166 | 85.45 | 77 |
| 47.943 | 1730 | 66.943 | 1051 | 85.943 | 645 | 47.95 | 248 | 66.95 | 149 | 85.95 | 100 |
| 48.443 | 1686 | 67.443 | 984 | 86.443 | 635 | 48.45 | 254 | 67.45 | 151 | 86.45 | 97 |
| 48.943 | 1676 | 67.943 | 1015 | 86.943 | 603 | 48.95 | 236 | 67.95 | 171 | 86.95 | 89 |
| 49.443 | 1641 | 68.443 | 1015 | 87.443 | 567 | 49.45 | 263 | 68.45 | 130 | 87.45 | 74 |
| 49.943 | 1682 | 68.943 | 952 | 87.943 | 636 | 49.95 | 274 | 68.95 | 159 | 87.95 | 96 |
| 50.443 | 1611 | 69.443 | 988 | 88.443 | 621 | 50.45 | 268 | 69.45 | 139 | 88.45 | 77 |
| 50.943 | 1603 | 69.943 | 964 | 88.943 | 634 | 50.95 | 246 | 69.95 | 138 | 88.95 | 75 |
| 51.443 | 1572 | 70.443 | 941 | 89.443 | 619 | 51.45 | 267 | 70.45 | 140 | 89.45 | 83 |
| 51.943 | 1476 | 70.943 | 874 | 89.943 | 579 | 51.95 | 261 | 70.95 | 136 | 89.95 | 68 |
| 52.443 | 1452 | 71.443 | 875 | 90.443 | 528 | 52.45 | 223 | 71.45 | 144 | 90.45 | 86 |
| 52.943 | 1449 | 71.943 | 875 | 90.943 | 570 | 52.95 | 225 | 71.95 | 130 | 90.95 | 70 |
| 53.443 | 1437 | 72.443 | 905 | 91.443 | 565 | 53.45 | 207 | 72.45 | 122 | 91.45 | 93 |
| 53.943 | 1456 | 72.943 | 840 | 91.943 | 593 | 53.95 | 235 | 72.95 | 115 | 91.95 | 88 |
| 54.443 | 1417 | 73.443 | 831 | 92.443 | 567 | 54.45 | 230 | 73.45 | 128 | 92.45 | 74 |
| 54.943 | 1359 | 73.943 | 825 | 92.943 | 546 | 54.95 | 203 | 73.95 | 121 | 92.95 | 72 |
| 55.443 | 1395 | 74.443 | 848 | 93.443 | 579 | 55.45 | 234 | 74.45 | 125 | 93.45 | 73 |
| 55.943 | 1278 | 74.943 | 890 | 93.943 | 547 | 55.95 | 205 | 74.95 | 99 | 93.95 | 80 |

| Saturation Mode | | | | Pulse Mode | | | |
|---|---|---|---|---|---|---|---|
| Time | Counts | Time | Counts | Time | Counts | Time | Counts |
| 94.443 | 552 | 230.443 | 473 | 94.45 | 48 | 230.45 | 44 |
| 94.943 | 528 | 235.443 | 482 | 94.95 | 71 | 235.45 | 44 |
| 95.443 | 517 | 240.443 | 439 | 95.45 | 75 | 240.45 | 36 |
| 95.943 | 510 | 245.443 | 420 | 95.95 | 72 | 245.45 | 28 |
| 96.443 | 529 | 250.443 | 397 | 96.45 | 66 | 250.45 | 23 |
| 96.943 | 488 | 255.443 | 387 | 96.95 | 49 | 255.45 | 43 |
| 97.443 | 476 | 260.443 | 297 | 97.45 | 68 | 260.45 | 19 |
| 97.943 | 484 | 265.443 | 288 | 97.95 | 62 | 265.45 | 20 |
| 98.443 | 472 | 270.443 | 284 | 98.45 | 68 | 270.45 | 22 |
| 98.943 | 499 | 275.443 | 263 | 98.95 | 66 | 275.45 | 28 |
| 99.443 | 474 | 280.443 | 243 | 99.45 | 62 | 280.45 | 23 |
| 99.943 | 496 | 285.443 | 224 | 99.95 | 52 | 285.45 | 17 |
| 100.443 | 453 | 290.443 | 222 | 100.45 | 54 | 290.45 | 25 |
| 105.443 | 4329 | 295.443 | 216 | 105.45 | 550 | 295.45 | 13 |
| 110.443 | 3819 | 300.443 | 199 | 110.45 | 476 | 300.45 | 19 |
| 115.443 | 3486 | | | 115.45 | 407 | | |
| 120.443 | 3227 | | | 120.45 | 354 | | |
| 125.443 | 2847 | | | 125.45 | 308 | | |
| 130.443 | 2584 | | | 130.45 | 321 | | |
| 135.443 | 2282 | | | 135.45 | 280 | | |
| 140.443 | 2122 | | | 140.45 | 234 | | |
| 145.443 | 2040 | | | 145.45 | 227 | | |
| 150.443 | 1875 | | | 150.45 | 184 | | |
| 155.443 | 1620 | | | 155.45 | 186 | | |
| 160.443 | 1526 | | | 160.45 | 155 | | |
| 165.443 | 1412 | | | 165.45 | 133 | | |
| 170.443 | 1228 | | | 170.45 | 115 | | |
| 175.443 | 1127 | | | 175.45 | 95 | | |
| 180.443 | 1055 | | | 180.45 | 90 | | |
| 185.443 | 1016 | | | 185.45 | 98 | | |
| 190.443 | 873 | | | 190.45 | 81 | | |
| 195.443 | 892 | | | 195.45 | 80 | | |
| 200.443 | 802 | | | 200.45 | 82 | | |
| 205.443 | 747 | | | 205.45 | 63 | | |
| 210.443 | 669 | | | 210.45 | 54 | | |
| 215.443 | 664 | | | 215.45 | 59 | | |
| 220.443 | 593 | | | 220.45 | 52 | | |
| 225.443 | 555 | | | 225.45 | 44 | | |

# VITA

Jinkai Wang was born in Hubei Province, China in 1975. He received his Bachelor of Science in nuclear engineering from Shanghai Jiao Tong University, China in 1997. After working in Guangdong Nuclear Power Station in China for 6 years, he entered the graduate program at Texas A&M University in August 2003 and received his Doctor of Philosophy in nuclear engineering in December 2008. His research interests include radiation detection and measurement, dosimetry evaluation, mathematical modeling and computational radiation protection and shielding.

Mr. Wang may be reached by writing to the Nuclear Science Center, 1095 Nuclear Science Rd., 3575 TAMU, College Station, TX, 77843-3575. His email is wangkincaid@hotmail.com.