nks

Nordisk kernesikkerhedsforskning
Norrænar kjarnöryggisrannsóknir
Pohjoismainen ydinturvallisuustutkimus
Nordisk kjernesikkerhetsforskning
Nordisk kärnsäkerhetsforskning
Nordic nuclear safety research

NKS-43
ISBN 87-7893-096-0

The Use of Synthetic Spectra to Test the Preparedness to Evaluate and Analyze Complex Gamma Spectra

Mika Nikkinen
Doletum Oy, Finland

October 2001

Abstract

This is the report of two exercises that were run under the NKS BOK-1.1 sub-project. In these exercises synthetic gamma spectra were developed to exercise the analysis of difficult spectra typically seen after a severe nuclear accident. The spectra were analyzed twice; first, participants were given short time to give results to resemble an actual emergency preparedness situation, then a longer period of time was allowed to tune the laboratory analysis results for quality assurance purposes. The exercise did prove that it is possible to move measurement data from one laboratory to another if second opinion of the analysis is needed. It was also felt that this kind of exercise would enhance the experience the laboratories have in analyzing accident data. Participants expressed the need for additional exercises of this type, this is inexpensive and an easy way to exercise quick emergency response situations not normally seen in daily laboratory routines.

Keywords

Gamma spectrum analysis; Quality assurance; Emergency preparedness; Synthetic data; Nuclear accident

NKS-43
ISBN 87-7893-096-0

Pitney Bowes Management Services Danmark A/S, 2002

The report can be obtained from
NKS Secretariat
P.O. Box 30
DK – 4000 Roskilde
Denmark

Phone +45 4677 4045
Fax +45 4677 4046
www.nks.org
e-mail nks@catscience.dk

Nordisk kernesikkerhedsforskning
Norrænar kjarnöryggisrannsóknir
Pohjoismainen ydinturvallisuustutkimus
Nordisk kjernesikkerhetsforskning
Nordisk kärnsäkerhetsforskning
Nordic nuclear safety research

01-10-01

The Use of Synthetic Spectra to Test the Preparedness to Evaluate and Analyze Complex Gamma Spectra

Report from the NKS/BOK-1.1 project

Mika Nikkinen
Doletum Oy
Finland

Table of contents

Summary

To mimic the situation on the real nuclear emergency, a new type of exercises was arranged under NKS/BOK-1.1 sub-project. In these exercises a set of spectra consisting of traces of severe nuclear accident were transmitted to number of laboratories for further analysis. The laboratories were asked to analyse the spectra as if they were internal samples from their own laboratories.

This kind of exercise with synthetic spectra has multiple benefits: the sample that contains radionuclides don't have to be transmitted to many laboratories, the problems with short lived radionuclides can be forgotten, the analysis results can be assumed to be promptly available and finally, the laboratories will receive spectra that they do seldom see in real operation, reminding that in case of emergency the spectra to be analysed could contain many things never seen in post Chernobyl era in most of the laboratories.

Two exercises were arranged, the first one autumn 2000 did contain leakage from VVER-440 type reactor (based on Origen calculation), whereas spring 2001 exercise did re-construct the Chernobyl nuclear accident as seen in Finland in May 1986. The first exercise was more simple, only 19 extra isotopes were added to the normal background spectrum. The Chernobyl sample did contain more than 30 extra isotopes as seen after the accident in Finland.

The exercises had emphasis on sample analysis needed for emergency preparedness purposes in case of nuclear accident. This kind of testing can be used also for in-house quality assurance. The exercise did prove that it is possible to share the measurement data in case of accident to have second opinion or to inform authorities abroad to foresee the situation in advance. For this purpose a conversion tool was supplied for participants to convert the spectra to a format better suitable for laboratory work.

In the two arranged exercises, an improvement was noticed in the analysis correctness, after prompt analysis in the first exercise 50% of the participants had good quality of results, in the second, more difficult case more than 75% of the participants did receive good results. The goodness of results was estimated by checking the number of correct isotopes identified and the accuracy of quantitative results reported by the laboratories. The participants were able to provide more comprehensive report after 2-3 week analysis phase. The correctness of results did improve after longer period of analysis time.

In case of emergency the time available for analysis is limited, the quality of analysis results should still be high enough for further decision. This is a dilemma which can be solved only by testing difficult cases in advance. The synthetic data can be used for this purpose to estimate when the quality of result is good enough and it is secure to release the results forward.

Introduction

Background

The NKS has previously organized laboratory intercomparison exercises, typically by distributing samples to be measured and giving at least some weeks for analyzing the samples. These exercises are useful for testing the performance of laboratories when measuring relatively long-lived radionuclides and when there is not strong time pressure (as e.g. in radioecology and various types of assessments). This does, however, not exercise well the type of performance needed for nuclear emergency preparedness. In the case of a nuclear accident, many laboratories would have to use gamma spectrometry for radionuclide identification and for quantification of these nuclides. This would involve the analysis of many radionuclides that the laboratories are not analyzing on a routine basis. A further complicating factor is that decision-makers would need the results very quickly.

The exercises discussed here were designed to address these special needs of nuclear emergency preparedness, but they were also meant to be of relevance for laboratories doing other types of gamma spectrometry (including radioecology). The exercises were performed under the NKS BOK-1.1 sub-project

Relevant test cases

In the exercises discussed here, synthetic gamma spectra were used in laboratory intercomparisons. This is the final report of the two different exercises, carried out in October 2000 and May 2001. The report contains descriptions of these exercises and discusses conclusions drawn from them.

It was decided that synthetic nuclear accident cases would be generated because in routine environmental analyses the laboratories will seldom see difficult cases. In the exercises realistic accident cases were developed and the laboratories were asked to analyze them as in real life. The cases were not published for the participants in advance, they were based on real inventories in a reactor (Exercise 1, Accident with VVER-440 type PWR reactor), and a real accident (Exercise 2, Chernobyl accident, RBMK reactor). Such cases could have to be dealt with by any laboratory in case of a nuclear accident.

It was assumed that the laboratories have to make quick analysis needed for further decision making. This would simulate a typical emergency preparedness situation, where correct answers are needed with fast response. In these kind of cases, the accuracy of the quantitative answers is not necessary the most important issue, 1 or 10 percent error estimate does not matter, but an order of magnitude or missing identification of a key radionuclide might cause a radiation hazard to be misinterpreted.

Since the intention was to improve internally the quality of laboratory analysis, the participating laboratories are only identified by numbers, not names, in this report.

Spectrum synthesis

The spectrum synthesis is an easy way to mimic a measurement situation that is difficult to set up or find in real life [1-12]. Usually the spectrum synthesis has following steps:

- Definition of isotopes and their concentrations
- Calculation of peaks associated with the isotopes
- Synthesis of peaks and Compton continuum
- Adding the Poisson statistics to the result to mimic the true nature of the measurement

Usually in the definition phase we have some kind of scenario, reactor accident, nuclear event or something else that is modeled well before the synthesis. Based on computer models like Origen, we can have quite relevant starting point for the synthesis.

The calculation of peaks is based on nuclear data or existing measurement data. Very comprehensive nuclear data are available, usually the gamma lines in the nuclear data are more comprehensive than is possible to see in normal measurement situations, since the lowest yield lines are below the detection limit.

The synthesis itself is based on peak and spectrum shape information (peak width, tailings, peak-to-Compton ratio) coming from some true detector measurement. The peaks and Compton continuum in this phase have clean shapes, what is still needed is to randomize the result a little bit to mimic the true nature of the measurement statistics.

Usually, we can make a distinction between three different synthesis methods:

- *Empirical method* is based on true measurement, there is no need to use any nuclear data library, the peaks in the original spectra are analyzed completely and the result peak table is the basement of the synthesis. The benefits in this method are that the peaks used here are really based on the real measurement, there are no missing and no extra peaks, also co-incidence-, sum- and escape peaks are treated automatically correctly. The organizer also knows the right peak areas and existing peaks in the spectrum correctly, therefore the evaluation of the results is easy. This was the case in the second exercise.

- *Semi-empirical method* is based on real measurement, a small signal (isotopes of interest and their Compton continuum) are added to the real measurement spectrum. This is a good method if we like to test the ability of different laboratories to detect small peaks (i.e. small concentrations of radionuclides of interest) from typical environmental surveillance spectra.

- *Completely synthetic method* is based on isotope activities known exactly before the synthesis. Then all the peaks and continuum are synthetic from the beginning to the end. This is easy to check as well, since the concentrations are known exactly. The problem in this case are the co-incidence-, sum- and escape peaks, if there is no model for these cases, these peaks will be missing from the synthetic data. This was the case in the first exercise, the escape-, sum- and co-incidence peaks were not added to make the analysis easier in the initial phase

Spectrum synthesis can also be a powerful tool for assessing the performance of detectors and planning improvements where needed. A user can visualize how his typical spectra would look like if he had a better detector. In some cases the better resolution of a detector makes it easier to analyze a spectra. Spectrum synthesis can be used by taking the sample spectrum from an existing detector, changing the resolution calibration and after the synthesis to figure out if this is a substantial improvement. It can also be used as a tool for selecting the correct detector type (NaI, CdTe, BGO, HPGe or some other) based on the measurement needs and sample information after the synthesis of the spectrum for each detector type. In some cases this can save substantial amount of money and result in considerable improvements of detection capability.

Objective of the exercises

The purpose of these exercises was to help the participants to evaluate their ability to analyze complex environmental sample spectra (from a HPGe detector), containing a variety of isotopes based on a nuclear accident. A synthetic spectra were generated containing a variation of fission products as seen in some reactor inventories (and assuming some cooling time). The spectra are based on true measurement of background, synthetic fission and activation product spectra were added. These kind of spectra are not usually seen during normal operations, but a number of laboratories should be able to analyze these types of spectra in case of an emergency.

The participants received in each exercise two spectra (the exercise spectrum + background spectrum) and calibrations for peak FWHM, energy and efficiency. Each participant was required to analyze the exercise spectrum and report the results at two different times (see appendix D for the time schedules of these exercises):

1. Promptly (preferably within 1 hour) after the spectra are received
2. Within one month after receiving the spectra (in each case a specific date was given.

The participating laboratories could decide themselves if they were going to give both of these reports. It was recommended that both of these reports should be given, even if the first analysis takes substantially more than one hour. The first report should be prompt for immediate emergency response, the second one is a more comprehensive one showing the capacity of the analysis team and methods.

Method of work

The synthetic spectrum used in each exercise is based on a true environmental spectrum. One scenario is used to generate additional artificial radionuclides. The artificial component is synthesized using the calibration information on the real spectrum (FWHM, peak tailings, energy, efficiency and peak-to-Compton ratio). Poisson statistic is added to the signal. The method makes the artificial spectrum very realistic. At this stage, the effect of random summing, escape peaks, and co-incidence summing are not included. This way the results based on peak detection, quantification and identification are easy to compare.

Following issues were examined during the test:

- Ability to detect small and medium size peaks in an environmental sample
- Ability to quantify the peak areas
- Ability to identify the isotopes present in the spectra
- Ability to quantify the activities of various isotopes in the spectra.
- Response time

The participants were expected in each case to deliver a report that contained at least:

- The information on detected peaks (energies, areas, intensities, error estimations)
- The information on detected radionuclides (nuclide names, activity and uncertainty estimations)
- Detection limits for a list of radionuclides (the list of isotopes is in appendix C)
- Spectra arrival time, the time when the analysis started and the reporting time

An example of such a report is included in appendix B.

The correctness of the results is easy to quantify since the added signal (pure peak areas, radionuclide concentrations and the detectable peak areas at the added signal) are well known by the organizer. Another component of the analysis is the timeliness; how quickly reliable result can be produced. This is very important from the emergency preparedness point of view. Another view to this exercise is the data interchangeability; how well the data measured by one institute can be processed and evaluated at another place.

The spectra were sent via email, the file format of the spectra was plain ASCII (one channel in one line, one calibration point-pair and error estimation in one line). The measurement, cooling and sampling times and sampled air volume (for an air filter sample) are included in the email. The spectra were also given in RMS2.0 ASCII file format and Ortec CHN format (note: pre-1997 format). The plain text file formats are provided in this document (appendix A).

How the correctness of results was estimated

The following items were used to quantify the correctness of the results:

- The number of correct peaks detected
- The number of correct isotopes identified
- The accuracy and the uncertainties of the results
- The timeliness of the answers
- The qualitative explanation of the events, i.e. interpretation of the spectra

Report on the first exercise

Scenario

This is the final report of the first data intercomparison exercise for NKS. In this exercise participants were asked to analyze synthetic spectra. The scenario was a reactor accident, 19 major isotopes typically seen in nuclear reactor inventories were added to the spectrum. The concentrations and isotopes were selected using Origen 2 calculations assuming average VVER-440 reactor inventory and 3 days of cooling time after the accident.

The added isotopes are listed in the following table, these activities are corrected for decay during counting:

Nuclide	Bq/sample
Zr-95	75.3
Zr-97	10.4
Nb-95	74.5
Mo-99	47.5
Ru-103	50.0
Rh-105	11.8
Te-132	38.2
I-131	34.5
I-132	39.4
I-133	18.1
Cs-134	0.73
Cs-136	0.67
Cs-137	1.58
Ba-140	70.3
La-140	77.0
Ce-141	71.9
Ce-143	26.4
Ce-144	35.7
Nd-147	25.7

To make the analysis easier, some peaks and some isotopes (as seen on Origen results) were dropped out from the synthetic spectrum. This was noted by one participant who had to force by hand I-132 to the identification solution due to missing peaks. The parent-daughter relationship was calculated by Origen. Therefore, for example the I-132 concentration is significantly different in the reports by the participants if the generation of I-132 during the sampling, decaying and data acquisition is not taken into account.

Because Poisson statistics is added to the spectrum, some of the added peaks are not visible in the spectrum and there might also be some variation in the reported activities.

Participants

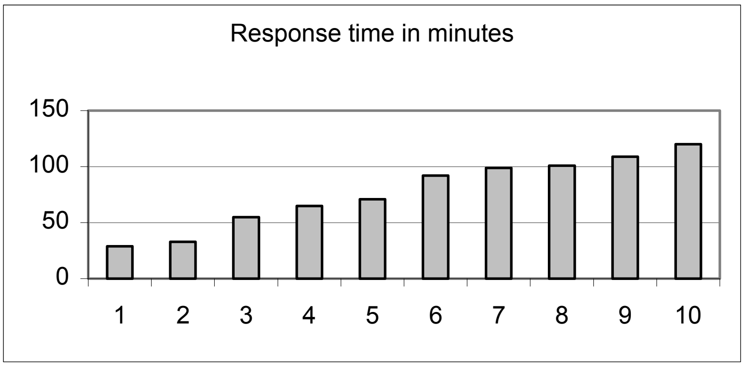
The spectra were sent to 25 laboratories in 9 countries. Laboratories from the following countries participated in the test:

Austria	1 participant
Denmark	4 participant
Estonia	1 participant
Finland	4 participants
Iceland	1 participant
Latvia	2 participants
Lithuania	3 participants
Norway	5 participants
Sweden	4 participants

Ten of these participants replied within the time limits of the first phase and finally altogether 14 replies were received within the longer evaluation period.

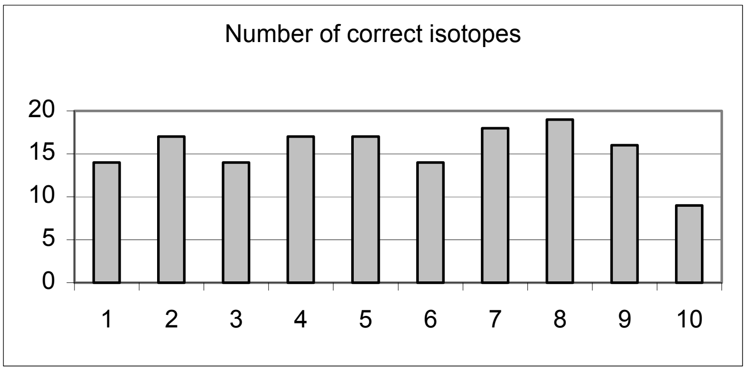
Quick response request, results

During the first phase the participants did not have too much time to consider the results, during the first 2 hours we received results from 10 participants, all of them showing good qualitative estimations of the composition of the spectra. The following graph is showing the response time in minutes for the participants sending the spectra during the first day.



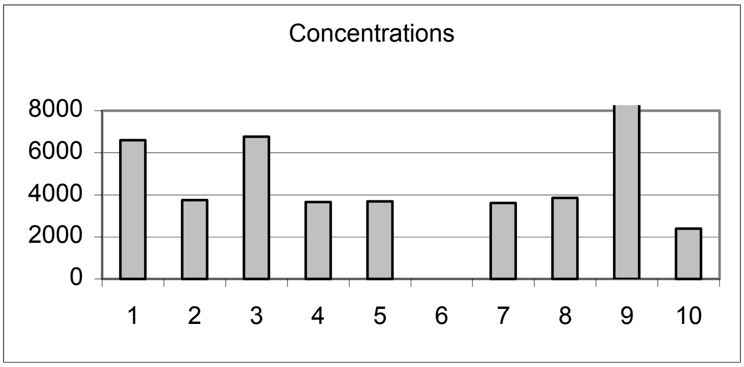
Two additional participants were replying later and two participants were sending final results or corrigendum to the results within one week.

The qualitative estimations of the results were good with all the participants. The following graph shows the qualitative identification results by the participants (10 reports within 2 hours):



Some variation in the activity calculations were noticed, one participant was reporting about 10 times too large activities, 2 participants had an error by factor 2 due to efficiency calibration problems and one participant was reporting too low activities probably due to missing decay correction.

One of the participants had limited the answers to qualitative only and one was reporting the activity of Te-132, only. Decay corrections were requested to be made with reference to the beginning of the counting period. Due to time limitations many participants were not able to change the correction scheme. All the participants were quoting the activity units clearly (Bq or Bq/m^3). Decay corrections were not quoted clearly. In the following graph, the concentration of Te-132 reported by participants is shown, five of the participants were rather close to the synthetic value (3.7 mBq/m3).



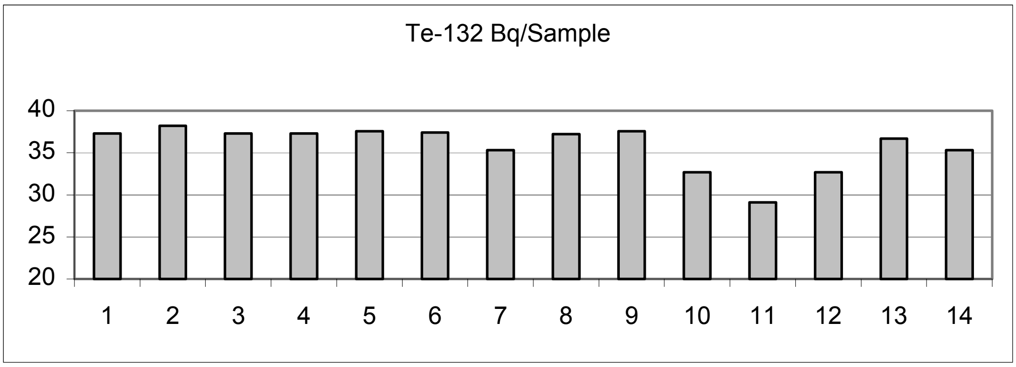
Final results of first exercise

There was about one month time to resolve the problem properly. During this time more accurate results were received. Finally 14 replies were received during the evaluation period. Some participants were not able to report results due to technical difficulties in converting spectrum data between different formats.

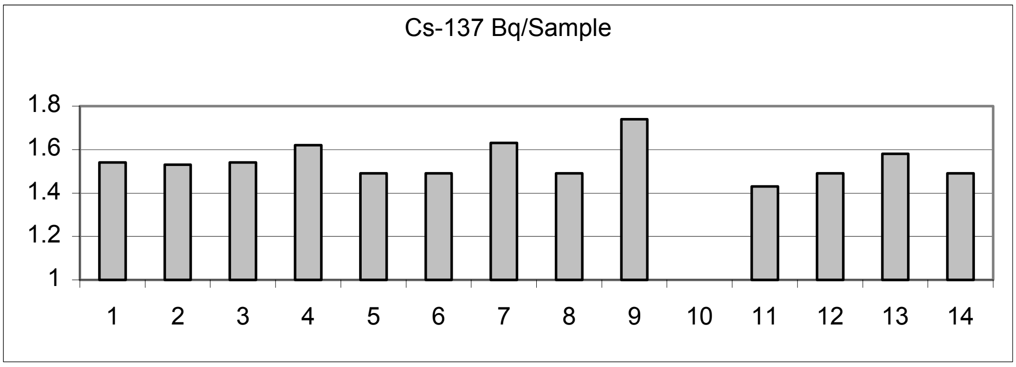
Correctness of activity estimations

We did receive some 4 additional replies and some corrections to the reported values. After corrective actions, all the participants were reasonably well within the right range.

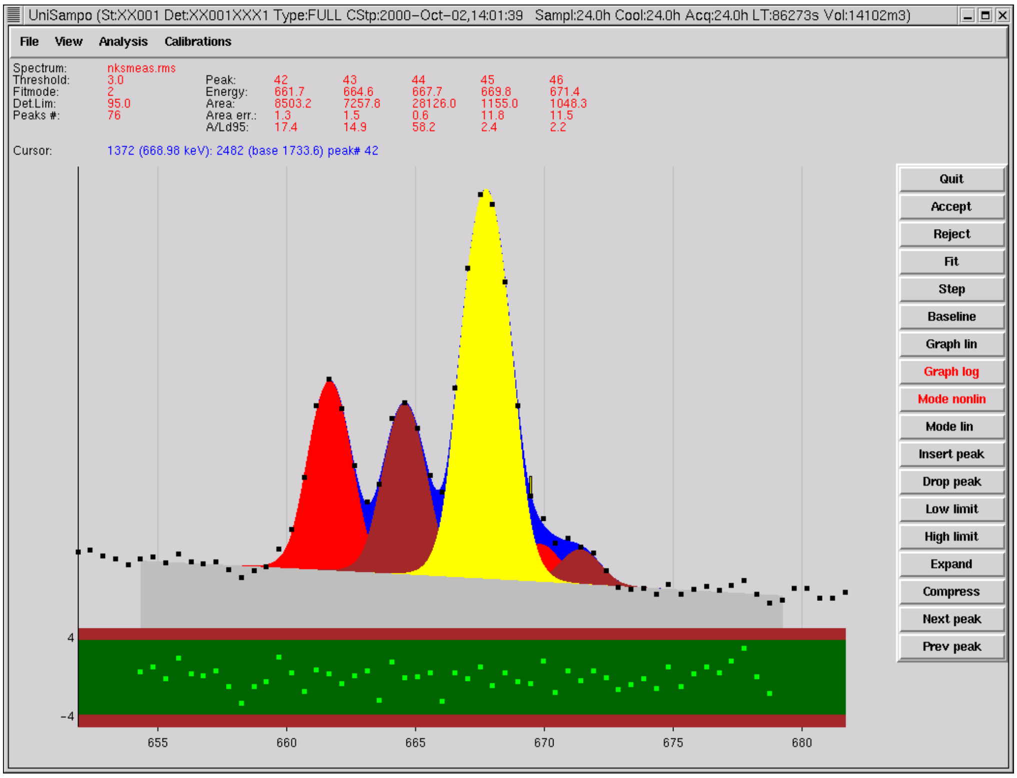
In the following chart, the activity of Te-132 is shown. The synthesized value was 38.2 Bq, Since a number of participants did report either Bq/m^2 with and with or without decay correction, a number of these activity estimations have been corrected to Bq/sample.



Cs-137 was also selected for the review, only one participant was not able to identify it. In this case we have interference of Ce-143 to the only peak of Cs-137. This caused some inaccuracy in the results. Synthesized value was 1.58 Bq. All the reported results were within 10% range of correct value.



In the following figure the actual peaks at the Cs-137 energy region are shown (UniSAMPO software [13-20]). The peaks belong to Cs-137, Ce-143 and the three latter ones for I-132.



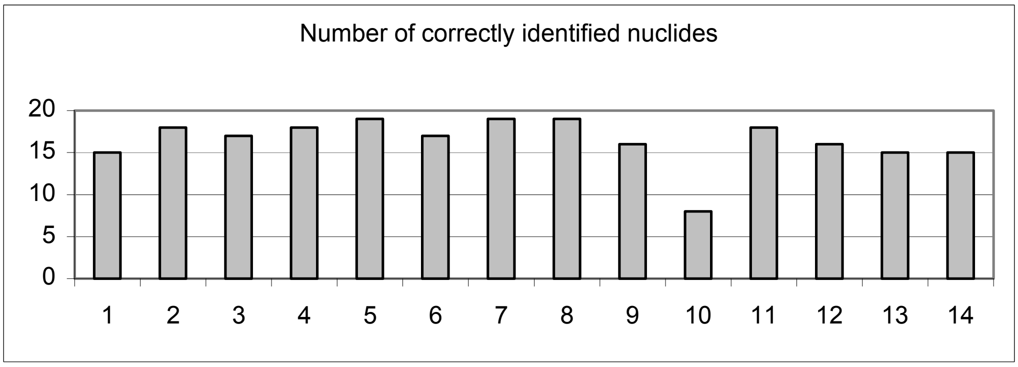
UniSampo (St:XX001 Det:XX001XXX1 Type:FULL CStp:2000–Oct–02,14:01:39   Sampl:24.0h Cool:24.0h Acq:24.0h LT:86273s Vol:14102m3)

```
Spectrum:     nksmeas.rms
Threshold:    3.0          Peak:       42       43       44        45       46
Fitmode:      2            Energy:     661.7    664.6    667.7     669.8    671.4
Det.Lim:      95.0         Area:       8503.2   7257.8   28126.0   1155.0   1048.3
Peaks #:      76           Area err.:  1.3      1.5      0.6       11.8     11.5
                           A/Ld95:     17.4     14.9     58.2      2.4      2.2

Cursor:       1372 (668.98 keV): 2482 (base 1733.6) peak# 42
```

Number of correctly identified isotopes

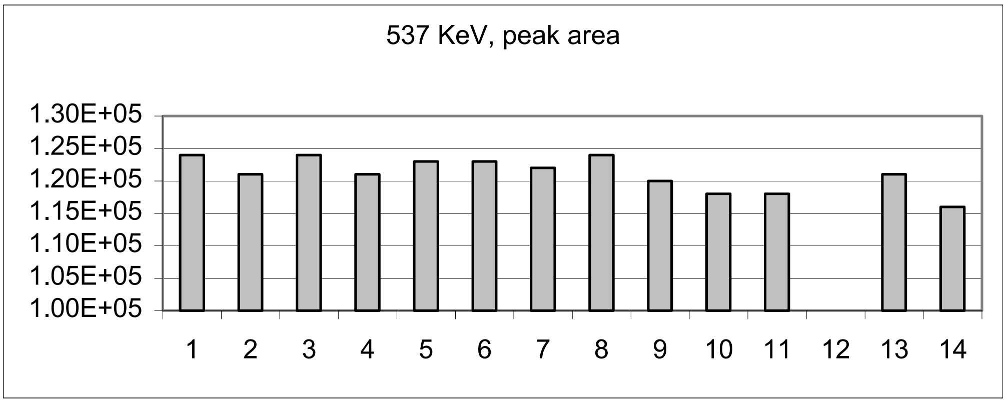
As in the original phase, the participants were reporting isotopes. The responses by participants were improved. Three participants were able to identify all 19 isotopes added to the spectrum. One participant had a somewhat limited gamma-line library. Usually the problem for the participants were Nd-147 (main peak outside energy range of the spectrum) and I-132 (activity estimation too large for identification, because parent, Te-132, was not taken into account by the analysis software). It should be noted that Tc-99m was not added to this spectrum, although many of the participants did identify it due to interference with Mo-99 (140.5 keV).



Accuracy of peak area definition, easy case

Since the synthesis was used to add a large amount of peaks to the spectrum, a number of peaks were below and close to detection limit. Therefore it is difficult to say how many peaks are really "visible" in the spectrum, the problem of analyzing small peaks is twofold: a true peak can be missed, and a non-peak can be claimed to be a real peak. In the typical processing some 70-80 peaks were found.

One medium size peak was selected for the investigations. The peak was the main peak of Ba-140 (537 keV). The response from participants was ranging from 1.16E5 to 1.24E5, the synthetic peak area (added peak) was 1.21E5. One participant did not report peak areas at all.

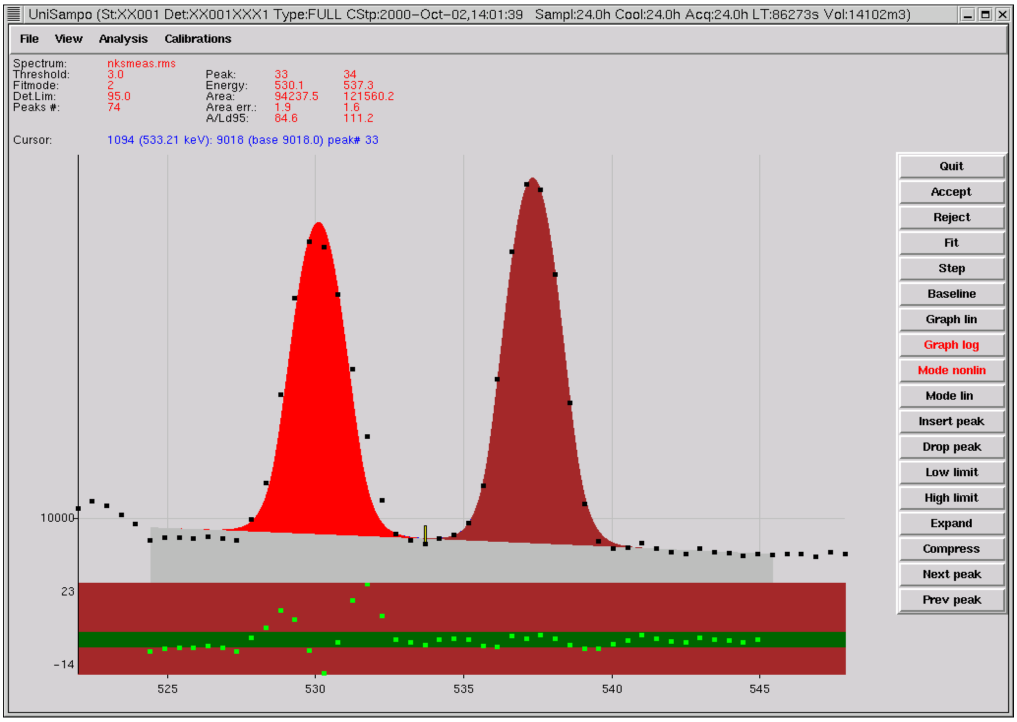


Accuracy of peak area definition, a small peak under a large one

This case seemed to be difficult to handle for most of the participants. Only one participant was able to give the right answer. The problem was to determine Nd-147 peak (531 keV) which was buried very badly by an I-133 peak (529.9 keV).

This multiplet is difficult to resolve automatically and it requires interactive processing. The user should be able to add peaks during this kind of processing, otherwise some very essential radionuclides might be missing from the results.

The following picture will show how this peak can be analyzed, In the first picture, the peak search has found only one peak in the 530 keV region. During the interactive analysis it is seen that the peak is not fitted properly. In the residual plot, a missing peak having significance of some 23 sigma is seen (UniSAMPO software).

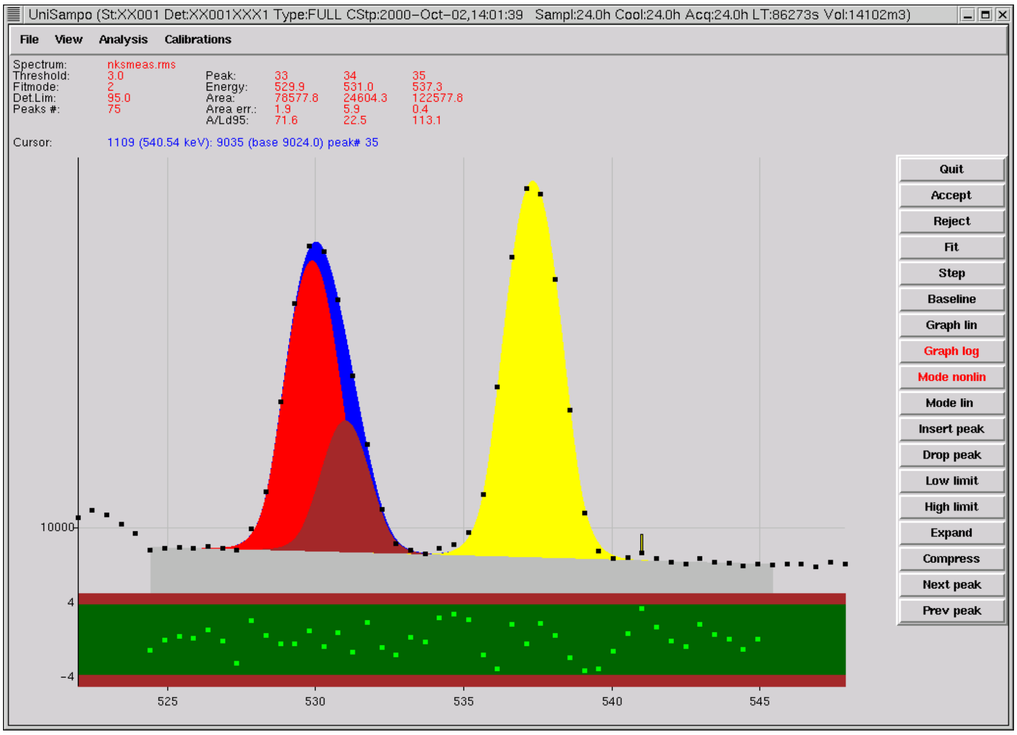


```
UniSampo (St:XX001 Det:XX001XXX1 Type:FULL CStp:2000-Oct-02,14:01:39   Sampl:24.0h Cool:24.0h Acq:24.0h LT:86273s Vol:14102m3)

Spectrum:      nksmeas.rms
Threshold:     3.0            Peak:        33          34
Fitmode:       2              Energy:      530.1       537.3
Det.Lim:       95.0           Area:        94237.5     121560.2
Peaks #:       74             Area err.:   1.9         1.6
                              A/Ld95:      84.6        111.2

Cursor:        1094 (533.21 keV): 9018 (base 9018.0) peak# 33
```

The analyst will add one peak close to the maximum residual, the analysis software is allowed to resolve the peaks using non-linear peak fitting (peaks are allowed to move to optimum position). After the fitting the peaks are on perfect position and the difference of the Nd-147 peak area compared to the synthetic one is about three percent (fitting uncertainty 5.9%). Here is the result:



```
UniSampo (St:XX001 Det:XX001XXX1 Type:FULL CStp:2000-Oct-02,14:01:39   Sampl:24.0h Cool:24.0h Acq:24.0h LT:86273s Vol:14102m3)

Spectrum:      nksmeas.rms
Threshold:     3.0            Peak:        33          34          35
Fitmode:       2              Energy:      529.9       531.0       537.3
Det.Lim:       95.0           Area:        78577.8     24604.3     122577.8
Peaks #:       75             Area err.:   1.9         5.9         0.4
                              A/Ld95:      71.6        22.5        113.1

Cursor:        1109 (540.54 keV): 9035 (base 9024.0) peak# 35
```

Technical problems

Technical problems were also encountered during the exercise, one problem was due to CHN file format, the file was based on pre-1997 format and was not completely Y2K compatible with newer analysis software. In the previous format year 2000 was presented as year 100 (1900 to be added to this year) but in the new format year 2000 should be presented as year 001 (00 is the year and 1 means that this is after the millenium change). Therefore we got (depending on software) year 1910, 2000 or 2010 to the field of start of counting.

File formats were also causing also other problems; the spectra were delivered in three different formats. However, a number of participants had difficulties in converting the spectra to individual systems. Specially, the calibrations (FWHM, Energy and Efficiency) were found to be difficult to import to some systems. There seems to be need for one universal data transfer format or for universal spectral data conversion software.

Another problem was encountered while opening the www page for file download. An interference between uploading the files to the server for distributing and requests for downloads at the same time from participants caused corruption of the exercise zip-file on the server and blocked the possibility of corrective action until 9:12 UTC. Thereafter access to the uncorrupted zip-file was OK.

Email data transfer was causing some problems as well, we did receive acknowledgements of the data reception 5 minutes to 1 hour after the spectra were sent (the transfer begun 09:00:05 UTC, the messages were completely transferred out from our own computer network 09:00:40 UTC). For the reference we were also sending one package to ourselves and it took 21 minutes to go to mail server and back. The cause of the delay was the large size of the messages and heavy load of Internet during the working hours.

Suggestions for future work

The spectra sent to the participants were completely synthetic, which makes the comparison of the results easy from the organizer point of view. The peak areas added to the spectrum are known accurately and therefore the quantification correctness is easy to calculate. This will be the method of work also in the future intercomparisons.

This case was quite easy, the concentrations at the spectrum were large enough that association of the isotopes was easy. The situation is different when the sample is much more active (kBq's or Bq's instead of mBq's per cubic meter of air) and when the concentrations are much closer to the detection limits. These cases will be tested in the following intercomparisons.

All the peaks from all the possible radionuclides should be synthesized. The participants should be able to quantify more difficult multiplets containing overlapping peaks. This capability is considered important by the organizer, our experience on real measurements is that we can have 250 – 400 peaks within 30-3000 keV energy range in a nuclear reactor based spectrum. Also summing and escape effects are essential in these cases.

Conclusions of the first exercise

As a conclusion of the results, in the quick response test 5 out of 10 got very good qualitative and quantitative results, three additional results were on the correct magnitude but contained large errors. The qualitative information was quite good already after few hours, but after one month the result was even better. One participant seems to have a more limited radionuclide library than the others.

Also the quantitative results improved significantly during the one month evaluation period, 11 participants were within 10% correct range in the estimation of the activity of Te-132. All of the participants were within 25% range. The most simple case, Cs-137 was following a similar pattern.

The quantification accuracy of simple large peak areas was found to be quite good, the main peak of Ba-140 was estimated within accuracy of 5% by all the participants. However, it should be noted that this peak was so large that the statistical error of this peak is 10 times better when using accurate methods. 5 of the participants were within 1% accuracy in this case.

The de-convolution of overlapping peaks seems to be a big problem. Only one participant was able to convolute the I-133/Nd-147 multiplet around 530 keV correctly. This multiplet needs interactive analysis but still it is reasonably easily solved if the analysis software is guiding the analyst during the analysis process.

Identification and activity calculation accuracy was eventually quite reasonable, some participants might want to review the results to see what is causing the problems in differences. Specially, importing external efficiency calibration was causing problems with a number of participants in the first (quick response) phase. It should be noted that participants that were able to read in directly all the data using IMS format (spectrum format including calibrations) were performing generally better than the others in this intercomparison.

The decay corrections were quite different from each other. By reading the report it was impossible to guess if the corrections were made or what was the reference date for the correction. During the evaluation three different sets of “similar results” were seen and the results seen in this documents were normalized assuming the results to be originated from one of these families (Bq/sample, μBq/m^3, μBq/m^3 without decay correction). This issue is to be emphasized in the future intercomparisons.

If a spectrum is transmitted to other institutions in an emergency, there should be either a universal data interchange format or all the institutions should have a conversion software which can be used to convert the spectra to individual systems. This is also considered to be important, this exercise was showing quite a lot of conversion problems with both spectrum and calibrations.

Report of second exercise

Scenario

In this exercise the scenario was based on the Chernobyl accident, the sample was re-constructed from a measurement of a swipe sample taken in Finland a few days after the Chernobyl nuclear accident. The sample was taken from a roof of one institute by swiping a glass panel, the measurement was performed in a laboratory without lead shielding, the sample was strong enough so that the substances on the sample dominated the spectrum.

In this synthetic case, the so called empirical method of synthesis was used. The original spectrum was carefully analyzed, peaks were quantified and the peak table was finally re-generated with spectrum synthesis. This way we know exactly what are the present peaks in the exercise sample spectrum. Altogether 245 peaks were added to the spectrum. They were all based on the original spectrum analysis. The peak table can be seen in appendix E. Because of the empirical method used, *all the peaks did also exist in the original spectrum,* this made the case very relevant since this kind of sample can be seen in real life in case of nuclear accident.

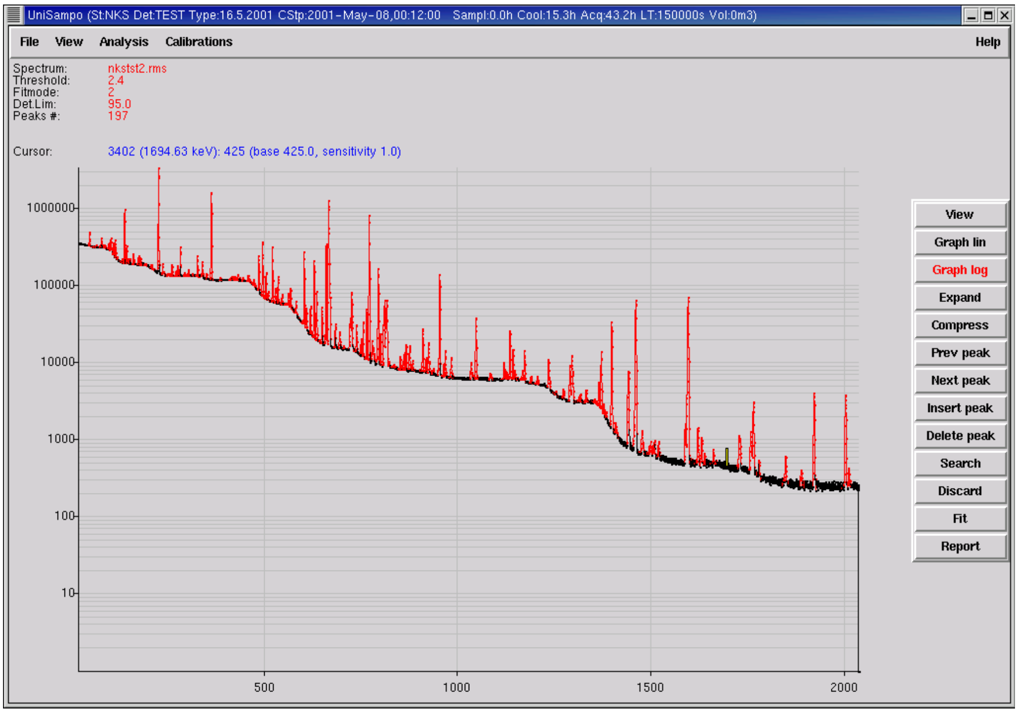
Some differences were seen in more rarely seen isotopes, some of the laboratories had performed really fine work, finding out explanation for nearly all peaks (good explanation for about 220 peaks).

The added isotopes are listed in the following table, the values are based on analysis of the initial peak table. Because of the method used, one cannot grant 100% accuracy for the concentrations, but this table gives one educated guess based on accurate analysis. Note that the natural radionuclides are present due to background (no lead shielding) and their activities presented in this table are not realistic because there has not been any background correction and we have used decay correction to the beginning of counting in this result table. These activities will result in a correct spectrum in case we assume the background to be zero.

Nuclide (+- 5-30%)	Bq/sample
Na-22	9.5
K-40	9750
Co-57	11
Co-60	22
Zr-95	235
Nb-95	230
Mo-99	940
Tc-99M	880
Ru-103	1140
Rh-105	21
Rh-106	330
Ag-110M	26
Ag-111	640
Cd-115	50
Sb-125	63
Sb-126	10
Sb-127	230
Te-129	1580
Te-129M	1040
Te-131	41
Te-131M	210
Te-132	6260
I-131	4700
I-132	7550
I-133	260
Xe-133	68
Cs-134	1070
Cs-136A	380
Cs-137	1900
Ba-140	1120
La-140	1320
Ce-141	230
Ce-143	22
Ce-144	160
Nd-147	58
Tl-208	110
Pb-212	170
Pb-214	100
Bi-212	580
Bi-214	390
Ac-228	440
U-235	16
U-237	23
Np-239	660

Exercise spectrum

The exercise spectrum had 4096 channels, the energy range was from approximately 30 to 2000 keV. It included a high number of fission and activation products. Measurement time for this spectrum was 150000s, so we see a lot of daughters coming in also during the counting.



UniSampo (St:NKS Det:TEST Type:16.5.2001 CStp:2001-May-08,00:12:00 Sampl:0.0h Cool:15.3h Acq:43.2h LT:150000s Vol:0m3)

Spectrum: nkstst2.rms
Threshold: 2.4
Fitmode: 2
Det.Lim: 95.0
Peaks #: 197

Cursor: 3402 (1694.63 keV): 425 (base 425.0, sensitivity 1.0)

Exercise spectrum, more than 200 peaks will be found with careful analysis.

Participants

The spectra were sent to 27 laboratories in 9 countries. Laboratories from the following countries participated in the test:

Austria	1 participant
Denmark	1 participant
Estonia	1 participant
Finland	8 participants
Iceland	1 participant
Latvia	3 participants
Lithuania	3 participants
Norway	3 participants
Sweden	6 participants

We received 18 reports from 8 countries. The following countries sent the report to the exercise organizer:

Austria	1 participant
Denmark	1 participant
Estonia	1 participant
Finland	6 participants
Iceland	1 participant

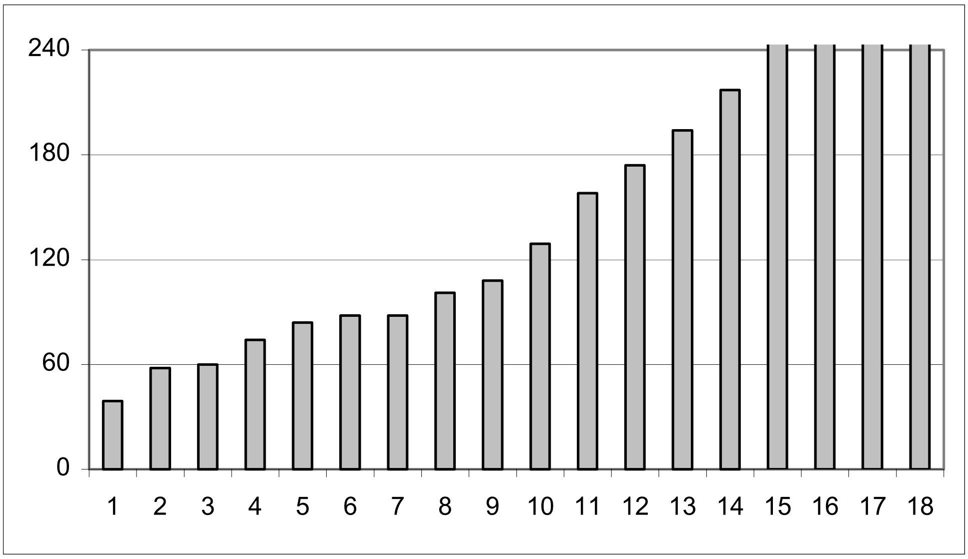
Lithuania	2 participants
Norway	2 participants
Sweden	4 participants

As in the previous exercise, during the first phase the participants did not have too much time to consider the results, during the first 4 hours we received results from 14 participants. One participant replied the following day because of computer problems and we did include the results of this participant into the "quick response" chart. We have received replies from 8 countries within 24 hours. The time to reply was typically longer than in the first exercise, a reason for this was the complexity of the spectrum. Figure 1 is showing the response time in minutes for the participants sending the spectra during the first day.

The quantitative estimation of the results was better than within the first exercise, some laboratories had still some technical problems in converting and importing the spectra. Due to new conversion software the reply percentage for the second round was 40% higher than in the first exercise (10 vs. 14).

Response time

Response time was reported already in the previous preliminary report, in addition to the first 15 replies during the first 24h after the delivery of the spectra, we received also 3 additional replies from the participants in the second phase. 11 out of 15 participants sent also a second reply, only 4 of the participants relied on the first results sent during the "quick" phase. So in the end we had 18 participants. 14 first replies were received roughly within 3.5 hours of the delivery of the spectra.

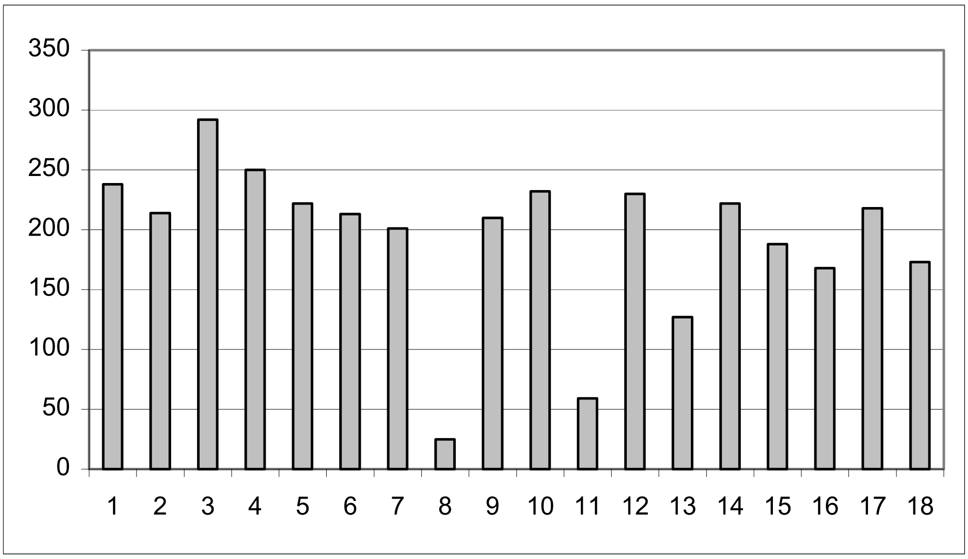


The response time in minutes for all the participants. Last 4 participants did not reply during the same day.

Results

We have added all the participants to the following charts, some of late participants did mimic "quick response" by sending the sample quickly when they were available and final results after a longer evaluation period.

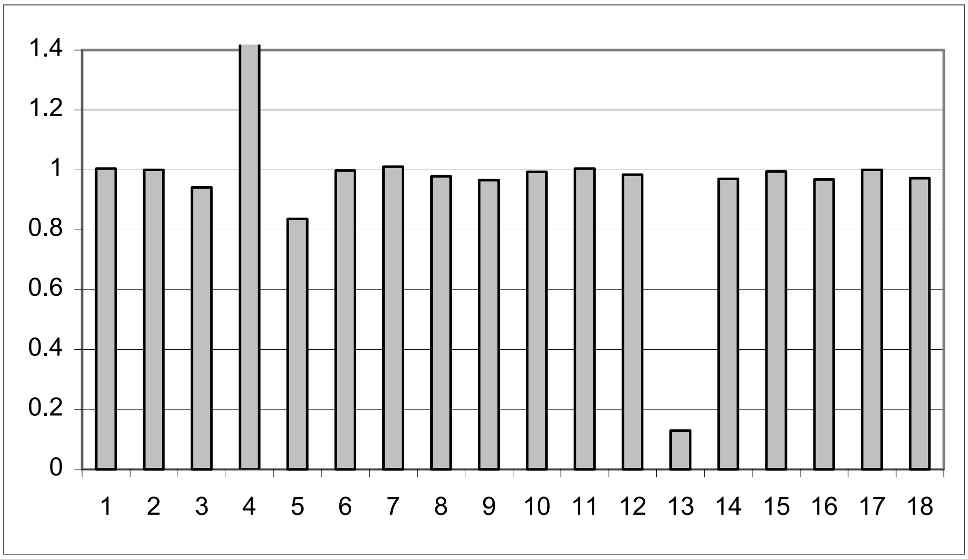
The number of peaks found after the longer evaluation period is shown in the following figure. Some of the participants used only library-oriented search and they reported only the peaks in their library or the found peaks and the library peaks together. This will probably explain the highest and lowest number of peaks in the following figure. 245 peaks were added to the spectrum in the synthesis.



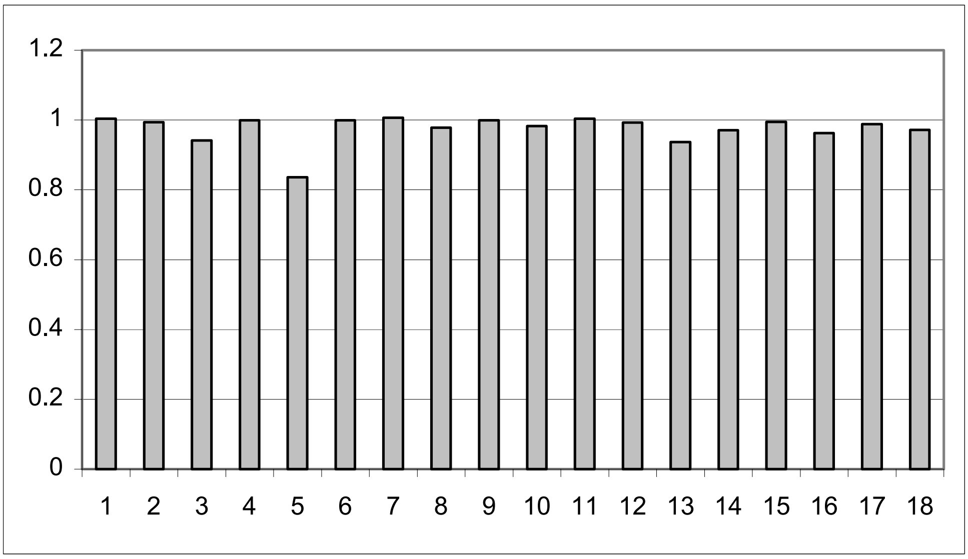
The number of peaks found is not the best and accurate way of detecting how well the participants found the peaks, in some cases the library oriented search can be dropping out non-valid candidates, whereas the traditional peak search method can too easily find spurious peaks.

Case 1, Cs-137. Two participants had problems within the first round. They did correct the result after the longer evaluation period. The Cs-137 case should be the easiest one due to no co-incidence problems etc. Some variation is seen because some neighboring peaks will harm the results

Like in the first exercise, some variation in the activity calculations was noticed. Some participants had problems resulting to large differences in expected values. For Cs-137, 11 out of 15 had results within calibration uncertainties, one was quite close to it, one had some 17 percent error and two participants had some bigger problems in analysis.

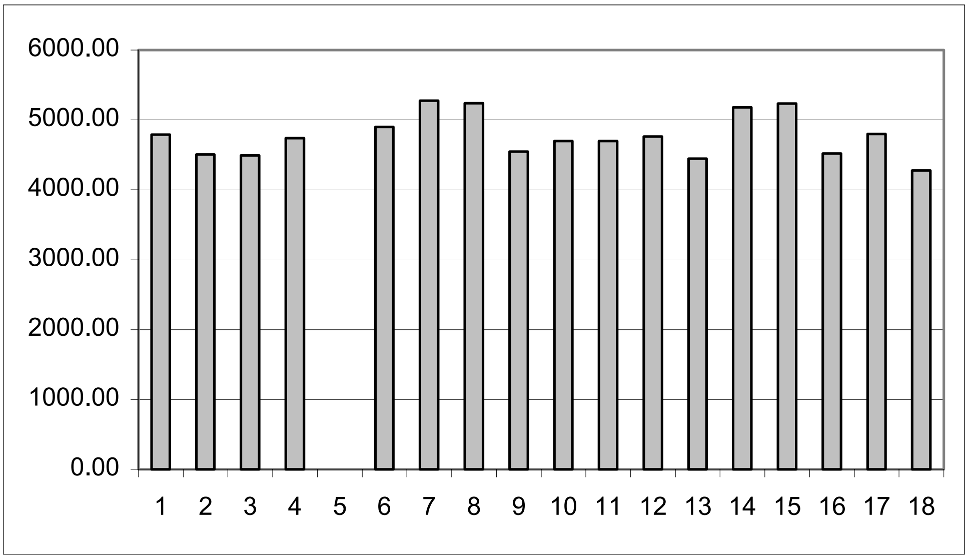


The result after “quick response time” 1.0 equals to exactly correct value.



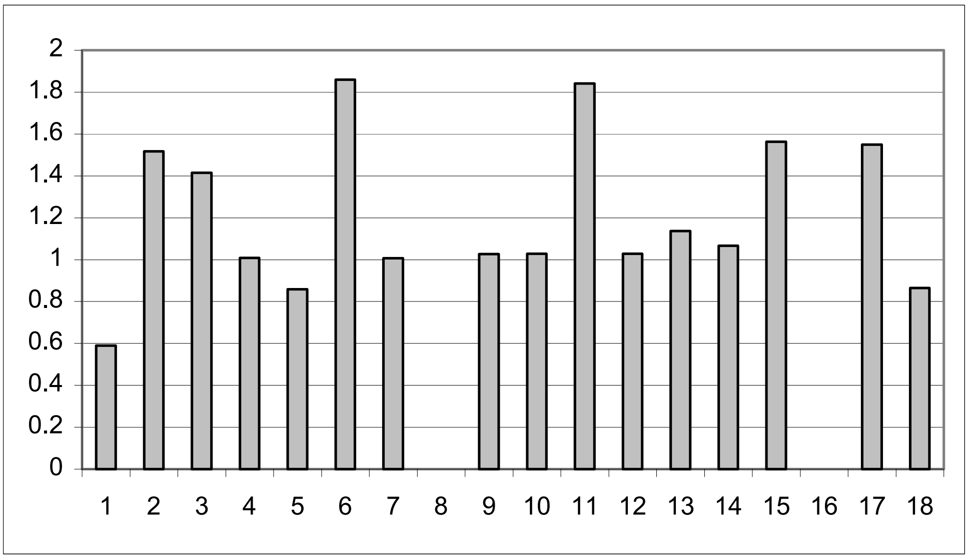
The result after long evaluation period. Nearly all participants were close to correct value. Only one is more than 10% from the expected value.

Case 2, I-131. Since iodine is probably most well known for its health effects, nearly all participants were close to correct value. One participant had probably some problems with decay correction or with reference dates, nearly all isotopes were correctly identified but especially the values of short-lived isotopes were far from expected values.

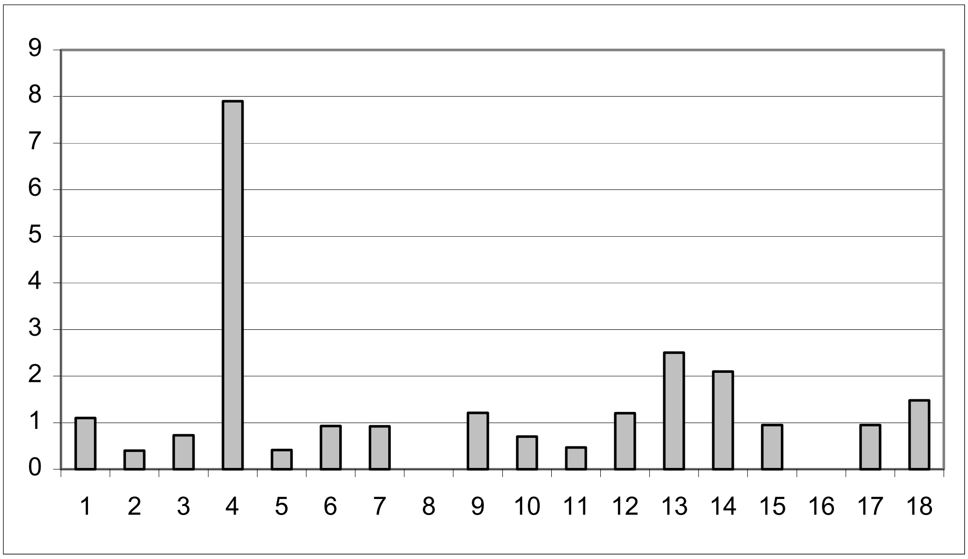


The I-131 concentration was expected to be around 4700 Bq.

Case 3. Peak at 724 keV, Zr-95, this is the second highest line of Zr-95. The known peak area was compared to the reported one. This peak is nicely buried with neighboring peaks based on, for example, iodine and cerium. If no de-convolution is used the peak area is incorrect by factor of tens percents. 6 out of 18 managed to de-convolute this case correctly, note that if the peak area is over-estimated because of merging two peaks around 723-724 keV together, it is not going to cause too large problems if the radionuclide identification process can associate the separate radionuclides to the same peak and if the activity calculation can handle LSQ method in finding out the correct activity for all the candidates explaining the same peak.

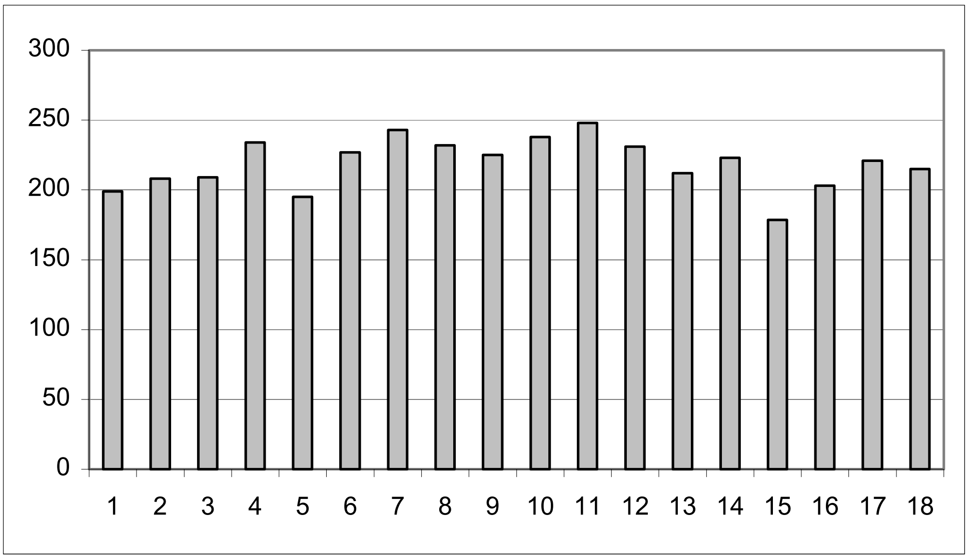


Zr-95 case, the reported peak area vs. the expected one. Two participants did not report peak areas or area errors at all.

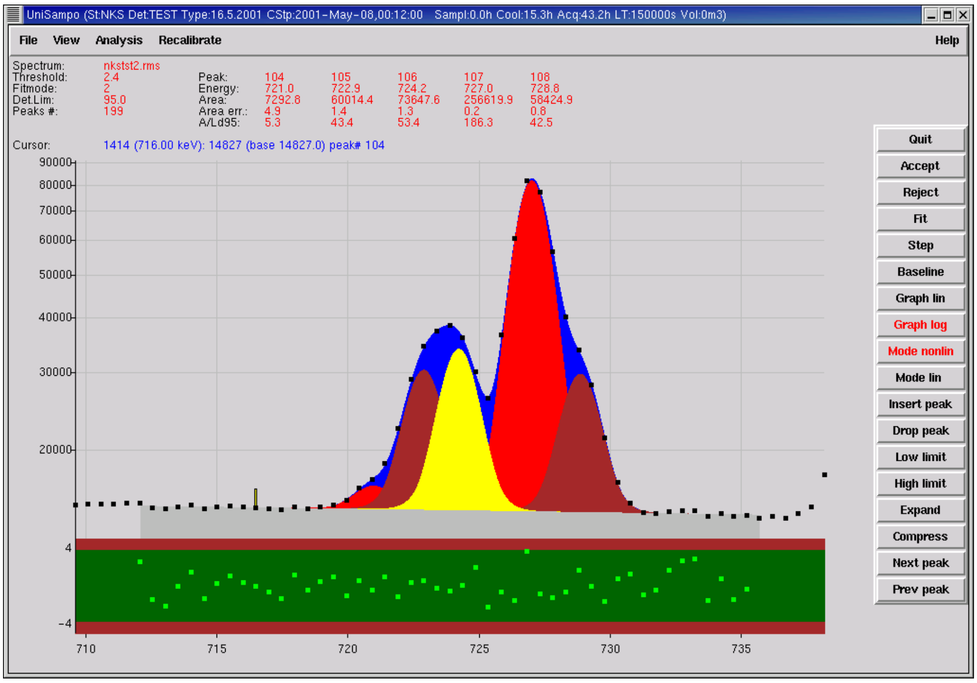


The error estimates reported by participants (in percentage) for Zr-95 peak areas.

Since there was another peak available at the energy of 756 keV, the uncertainties of this peak did not cause total breakdown in the activity calculations. However, the variation of the activity was quite large if we take into account that these two peaks used for the identification were quite large (statistical peak area error around 1%).



The activity of Zr-95. The expected value was 235 Bq, the variation was quite large if we take into account that typical peak area errors reported by the participants were around 1%.



```
UniSampo (St:NKS Det:TEST Type:16.5.2001 CStp:2001-May-08,00:12:00   Sampl:0.0h Cool:15.3h Acq:43.2h LT:150000s Vol:0m3)

Spectrum:    nkstst2.rms
Threshold:   2.4       Peak:       104      105      106      107       108
Fitmode:     2         Energy:     721.0    722.9    724.2    727.0     728.8
Det.Lim:     95.0      Area:       7292.8   60014.4  73647.6  256619.9  58424.9
Peaks #:     199       Area err.:  4.9      1.4      1.3      0.2       0.8
                       A/Ld95:     5.3      43.4     53.4     186.3     42.5

Cursor:      1414 (716.00 keV): 14827 (base 14827.0) peak# 104
```

This is a picture of correct de-convolution of Zr-95 (724 keV), Zr-95 peak is the one in the middle (yellow), the other peaks are due to Sb-126 (721 keV), Sb-127, I-131 (723 keV), I-132, Bi-212, Ac-228 (727 keV) and I-132, Te-129M (729 keV).

Technical problems

Technically, the exercise was performed better than the first one, there were still some problems, but the situation was easier than in the first exercise.

File formats were still causing some problems, this time the plain ASCII files were not sent to the participants, instead the conversion software was supplied in the data package. Some assistance was given in describing the usage of this conversion software for some participants. One problem were still the users of Canberra Genie, the format of Genie (CNF) is closed and Canberra did not give out the format for us. We will negotiate with Canberra on possibilities to include their formats also into the conversion software package.

Email data transfer was working well this time. The transfer began 08:59:00 UTC, the messages were completely transferred out from our own computer network at 08:59:25 UTC. For the reference we were also sending one package to ourselves and it took 5 minutes to go to the mail server and back. In the last exercise the same transfer process took 21 minutes due to the large size of the messages and heavy load of internet during the working hours.

Downloading from the BOK-1.1 web site worked well this time, no problems were encountered by the organizers and no problems were reported by the participants. This time the files were placed on the web site before the start of the exercise and access provided to them by adding a link from a special Web page at 09:00:00 UTC.

Suggestions for future work

Also in this exercise the spectra sent to the participants were completely synthetic. This makes the comparison of the results easy from the organizer point of view. The peak areas added to the spectrum are known accurately and therefore the correctness of results is easy to calculate. This time the synthetic spectrum was made using the real measurement that was carefully analyzed, peaks quantified and the result of this peak analysis was synthesized. The case was real, it was based on a truly relevant case, as described in this report.

This case was substantially more difficult than the previous one, there were well over 200 peaks and the participants had some problems in explaining the radionuclides. This is a case that could be on a desk of the participants in real life, since the origin of the spectrum is based on one real case.

It is suggested that one more exercise with synthetic spectra be carried out. In this exercise the calibration spectrum would be given one week before the exercise and the participants might calibrate the analysis software in time. In this case the situation would mimic a case where the laboratories had measured the exercise spectrum with their own detectors. Also the support for Canberra CNF files would be available, this would give some laboratories better possibilities to participate in the exercise.

Remarks of the second exercise

The second exercise was more demanding than the first one. This was seen in the reply time. Because of the number of peaks, the analysis was much more demanding. The case was selected so that all the peaks seen in the spectrum were relevant, incoming daughters made the situation quite interesting, specially, since the sample was active enough.

There were still some technical problems that caused some participants to be unable to send the results. One was related to file formats: Canberra Genie seems not to have a convenient way to import plain ASCII spectra files. This issue should be taken into consideration if a similar exercise is arranged again. Another issue is that some laboratories are concentrating on their own detectors and do not possess tools to import calibrations from other sources. This issue can be overcome in the future if the calibration spectrum is transmitted to the participants well before the exercise. In this case the situation would be equal to one where the measurements had been done at the local laboratory.

The Cs-137 example shows that in simple cases, nearly all participants are able to reach +-5% accuracy. This was expected, although some problems were seen in the efficiency calibration and spectral data conversion in two cases. The I-131 case shows that even large errors can be possible, but nearly all participants were able to receive right order of magnitude. 17 of the participants were within the range of +-20% of the expected value. The Zr-95 case shows that not all laboratories are able to de-convolute difficult multiplets. The case shows that the error estimates tend to be underestimated in such cases.

The intercomparisons material consists of a number of interesting issues that can be checked and compared against one another. Because the sample was so strong, we did not put too much emphasis on detection limits, nearly all radionuclides of interest were present in the spectrum. We did note, however, that some participants lost some very relevant isotopes in the analysis phase. This exercise can be used as internal quality control to check the behavior of the system in quick-response type analysis and in more detailed, interactive analysis.

Spectrum Conversion Software for Data Intercomparison Exercises

Introduction

In the first NKS exercise with synthetic spectra it was discovered that not all laboratories were able to convert the spectra to their own data systems. Usually the problem was limited ability of the analysis software to read in data in other format than its own. The spectra were transmitted in three different formats: RMS2.0, Ortec CHN and plain ASCII. The RMS2.0 format includes also calibrations, with CHN and ASCII formats the users needed to import the calibrations listed in separate files.

The absence of conversion software caused some laboratories to cancel the participation to the exercise. Therefore it was decided that a general purpose conversion software was to be developed before the second exercise with synthetic spectra (May 2001).

New Software

The new conversion software did initially contain 4 different formats: RMS2.0, Ortec CHN, Ortec SPC and plain ASCII. Conversion to Canberra CNF files was discussed also but the file format specification was not available for conversion software development. According to discussions (August 2001) with Canberra, the file format may be available, but it is unclear if it requires proprietary libraries or copyrighted material, which cannot be used with this kind of free software.

One idea of the conversion software is that the software is completely free. The users may download source code, modify it, add new file formats and distribute the software freely among the colleagues. It is planned that the software will be distributed even under GNU public license (GPL), which allows free distribution of the software, modifications, source code to be included but the software must be distributed freely, without license fee. This would allow wide user group to use and develop the software. GPL is very widely used in the field of free software, for example, the Linux operating system is distributed under this license.

The Usage of the Software

The usage of the software is easy. The usage is printed out if command convspe is given:

```
CONVSPE spectrum file converter. Version 1.05
usage: convspe -type inputfile -type outpufile

type may be:

       -rms (RMS 2.0 file format)
       -asc (Plain ASCII file, one channel on each line)
       -chn (Ortec CHN file format)
       -spc (Ortec SPC file format)
```

So for example to convert RMS 2.0 format spectrum file to Ortec CHN format, use command:

```
convspe –rms rmsfile.rms -chn chnfile.chn
```

Notice that only SPC and RMS file formats include calibrations, plain ASCII and CHN file does not include any information on energy, peak FWHM or efficiency calibration point pairs.

Notice also that SPC file, while being comprehensive in general, does not include error estimates for calibrations. This means that some default calibrations has to be used

Future work

The software is free to receive additional file formats, anybody able to write C-programming language can add new formats to this package. One has to introduce first the read in and write out routines compatible to convspe data structure, then add read in entry and write out entry to main.c file. Compile the c files and use it.

The source code of the spectrum conversion software is given in Appendix F. The current version (1.05) has following C-language files:

Main.c	Main program, includes command line procedures
Spc.c	Spc file format read in and write out commands
Rms.c	Same for RMS 2.0 and ASCII files.
Chn.c	Same for CHN files
Times.c	General time routines
All.h	All structures and data blocks needed by convspe.

World is full of different kinds of spectrum data formats, this is one way to manage the ability to convert from one to another format. We will welcome any voluntary contributions to enhance the capability of this conversion software. The source code is available on request, email: Doletum@kolumbus.fi.

Conclusions

The exercises showed that even the "most simple" phase in the gamma spectrometric analysis, the data analysis, can be a source of uncertainties. The exercises helped to identify many interesting issues that the participants have found out themselves from their analysis systems. It is a very good idea from time to time to test the system with more difficult cases to see if everything is working well and if the operators do still remember what to do in difficult cases like these.

These cases were both relevant nuclear accidents for two different reactor types. We have seen these cases in real life in the past, but since the staff is changing and experts are in some cases approaching pension age, these kinds of exercises would give nice thrill for a new generation of gamma analysts.

The idea of these exercises was to show how the situation can be if the laboratory has to work under pressure in a real situation, time is luxury that is not necessarily available, the problems are new and unexpected, the peaks are in the positions where they should not be, one might argue if the calibrations are really correct, this sample cannot be really this hot, can it? There are questions about responsibilities, specially accuracy vs. time to use to find out the perfect solution. The next step could be to have a rehearsal on a realistic case where the staff has to make decisions, also from a radiation protection point of view, there is not only one sample to analyze, similar cases are floating in the whole afternoon, laboratory will receive multiple samples per hour containing high radiation levels. Some technical problems will occur in the laboratory, and there is no time to call maintenance to fix the situation.

The exercises showed that gamma spectra can be exchanged effectively between the participating laboratories. A special software was developed that enables users to translate spectral data from one format to another (this is described in a separate NKS/BOK-1.1 report). The main difficulties in exchanging between different file format lies in the proprietary Canberra Genie file format. Information about its structure is not openly available and thus it could not be covered by the special conversion software. Canberra has been contacted because of this issue and hopefully a solution can be found.

The exercises showed that the Internet can be used efficiently to distribute exercise data to participants at a defined time, provided care is taken to use efficient servers and to avoid conflicts arising from participants trying to download information from a server while it is being uploaded there. No flaws were detected in the use of the Internet in the second exercise.

Some improvements were seen between the exercises, the quick responses were better within the correct range in the second exercise. There are still some issues that need to be addressed with care in many laboratories. Typical problems in the sample data analysis are concentrated to the following areas:

- Location and quantification of small peaks
- De-convolution of overlapping peaks
- Activity calculation in case of interferences and co-incidences
- Accuracy of the efficiency calibration

The exercise has been a pleasant experience for the organizer. The participants provided stimulating feedback during and after the exercises, including suggestions for items that could be included in possible future exercises. One such item is to distribute calibration information well in advance so that users could fine-tune their systems to the test spectra before the exercise. This would mimic better a situation where the spectra are generated by the user's own system.

When planning a possible future exercise the following needs to be considered:

- Define the objective of the exercise clearly; if the emphasis is more on emergency preparedness, the participants should concentrate on prompt replies more than in the previous exercises.

- What is the benefit if one is arranged; a new set of test spectra could be a baseline for future in-house training of the participating laboratories, the analysis system could be tested for quality assurance purposes specially in case when a new software package is procured or upgraded;

- What are the potential risks if these kinds of cases are not practiced and an accident will occur

The organizer would like to encourage laboratories to study what they feel they have learned from this set of exercises and then to send suggestions for possible future work to the NKS. The organizer would also be quite happy to receive suggestions and comments from the participating laboratories in order to help to develop a new plan for exercises of this type.

Acknowledgements

The management of the exercises on behalf of NKS BOK-1.1 was excellent, many thanks to Sigurður Emil Pálsson and Geislavarnir ríkisins in setting up and maintaining the WWW site for the exercise. The organizer would also like to thank Bent Lauritzen, the project leader of BOK-1, for the seamless co-operation during the exercise.

References

1. R. Hulmi, Gammaspektrien synteesi simuloiduilla Ge(Li)-detektorin vastefunktioilla. Diplomityö. Teknillinen korkeakoulu, Teknillisen fysiikan osasto 1979 (in Finnish).

2. T. Tynkkynen, Gammaspektrien syntetisoinnin viimeistelyä. Erikoistyö, Ydin- ja energiatekniikka. Teknillinen korkeakoulu, Teknillisen fysiikan osasto 1981 (in Finnish).

3. J. Laurimaa, Gammaspektrin syntetisointiohjelma mikrotietokoneympäristöön. Erikoistyö, Tfy-56.198 Ydin- ja energiatekniikka. Teknillinen korkeakoulu, Tietotekniikan osasto 1990 (in Finnish).

4. P.A. Aarnio, M.J. Koskelo, "OPTO: Optimization Program for Multicomponent Activation Detectors". Nuclear Technology 59 (1982) 170-185.

5. Y. Jin, R.P. Gardner, K. Verghese, "A Semi-Empirical Model for the Gamma-Ray Response Function of Germanium Detectors Based on Fundamental Interaction Mechanisms". Nuclear Instruments and Methods in Physics Research A242 (1986) 416-426.

6. M.C. Lee, K. Verghese, R.P. Gardner, "Extension of the Semiempirical Germanium Detector Response Function to Low Energy Gamma Rays". Nuclear Instruments and Methods in Physics Research A262 (1987) 430-438.

7. D.J.G. Love, A.H. Nelson, "Unfolding the Response Function of High-Quality Germanium Detectors". Nuclear Instruments and Methods in Physics Research A274 (1989) 541-546.

8. J.C. Waddington, "Comment on "Unfolding the Response Function of High-Quality Germanium Detectors" by Love and Nelson". Nuclear Instruments and Methods in Physics Research A274 (1989) 608-609.

9. D.J. Mitchell, H.M. Sanger, K.W. Marlow, "Gamma-Ray Response Functions for Scintillation and Semiconductor Detectors". Nuclear Instruments and Methods in Physics Research A276 (1989) 547-556.

10. C.E. Moss, J.R. Streetman, "Comparison of Calculated and Measured Response Functions for Germanium Detectors". Nuclear Instruments and Methods in Physics Research A299 (1990) 98-101.

11. W.K. Hensley, A.D. McKinnon, H.S. Miley, M.E. Panisko, R.M. Savard, "SYNTH: A Spectrum Synthesizer". Journal of Radioanalytical and Nuclear Chemistry 193 2 (1995) 229-237.

12. T. Kishikawa, S. Noguchi, C. Yonezawa, H. Matsue, A. Nakamura, H. Sawahata, "Photopeak Prole of Full Energy and Escape Peaks in Neutron Capture Prompt-Gamma Ray Spectra". Journal of Radioanalytical and Nuclear Chemistry 215 2 (1997) 211-217.

13. J.T. Routti, Sampo, “A Fortran IV Program for Computer Analysis of Gamma Spectra from Ge(Li) Detectors”. University of California, Lawrence Berkeley Laboratory”, Report UCRL-19452 1969.

14. J.T. Routti, S.G. Prussin, “Photopeak Method for the Computer Analysis of Gamma Ray Spectra from Semiconductor Detectors”, Nuclear Instruments and Methods, 72 (1969) 125.

15. P.A. Aarnio, J.T. Routti and J.V. Sandberg, “MicroSampo - Personal Computer Based Advanced Gamma Spectrum Analysis System.”, Journal of Radioanalytical and Nuclear Chemistry, 124 2 (1988) 457.

16. M.J. Koskelo, P.A. Aarnio, J.T. Routti, Sampo80: “An Accurate Gamma Spectrum Analysis Method for Minicomputers”. Nuclear Instruments and Methods, 190 (1981) 89.

17. M.J. Koskelo, P.A. Aarnio, J.T. Routti, Sampo80: “Minicomputer Program for Gamma Spectrum Analysis with Nuclide Identification”. Computer Physics Communications, 24 (1981) 11.

18. P.A. Aarnio, M.T. Nikkinen, J.T. Routti, “High Resolution Interactive Gamma Spectrum Analysis Including Automation with Macros”, Journal of Radioanalytical and Nuclear Chemistry, 160 (1992) 289.

19. P.A. Aarnio, M.T. Nikkinen, J.T. Routti, “Gamma Spectrum Analysis Including NAA with Sampo for Windows”, Journal of Radioanalytical and Nuclear Chemistry, 193 (1995) 179.

20. P.A. Aarnio, M.T. Nikkinen, J.T. Routti, “UniSAMPO, comprehensive software for gamma spectrum processing”, Journal of Radioanalytical and Nuclear Chemistry 248 2 (2001) 371-375.

Appendices

Appendix A - File formats for the spectra

The spectra are transferred in ASCII format; two possible formats are available:

1. Plain ASCII files, multiple files with same base name:
   Spectrum: extension .sp
      1 channel per line
   FWHM Calibration: extension .fw
      Channel, FWHM and error for FWHM on each line
   Efficiency calibration: extension .ef
      Energy, efficiency and error for efficiency on each line
   Energy calibration: extension .en
      Channel, energy and error for energy on each line
   Spectrum info: extension .inf
      Time information (sampling, cooling, acquisition, live, real times)
      Sampled air quantity
      Other information, if needed

2. RMS 2.0 (or IMS 1.0) ASCII file format, all calibrations and spectrum in single file
   The data is stored ordered according to keywords, following keywords are specified:

   #Collection
      On single line:
      - Start of air sampling date and time
      - End of air sampling date and time
      - Quantity of the air (in m^3)

   #Acquisition
      On single line:
      - Start of acquisition date and time
      - Real measurement time (s)
      - Live measurement time (s)

   #Energy
      On each line after the keyword
      - energy (in keV)
      - channel
      - error (energy)

   #Resolution
      On each line after the keyword
      - energy (in keV)
      - FWHM (in keV)
      - error (FWHM)

#Efficiency

On each line after the keyword
- energy (in keV)
- efficiency
- error (efficiency)

#Spectrum

First line: number of channels and optional energy calibration
On each line after the keyword until the end of file or STOP
- Channel number
- The contents of next 5 channels

Here is an example on this kind of file type:

```
BEGIN RMS2.0
MSG_TYPE DATA
MSG_ID 43456
DATA_TYPE SAMPLEPHD
#Header
KW001 KW001KWA1 P DISK FULL
1998/07/04 08:33:32.0
#Comment
rashad - Detector 1 Ranger (3-JUL-1998)
#Collection
1998/07/02 10:44:58.9 1998/07/03 12:44:30.4 15290.00
#Acquisition
1998/07/03 13:02:55.3 70200.0          70185.7
#Energy
88.000000       179.310806       0.100000
122.099998      248.938995       0.100000
158.899994      324.350525       0.100000
391.700012      800.386902       0.100000
514.000000      1049.999878      0.100000
661.700012      1352.871216      0.100000
898.000000      1836.612305      0.100000
1173.199951     2400.183350      0.100000
1332.500000     2726.339844      0.100000
1836.099976     3757.629883      0.100000
#Resolution
88.000000       1.947902         0.100000
122.099998      2.018639         0.100000
158.899994      1.933774         0.100000
391.700012      2.141653         0.100000
514.000000      2.435176         0.100000
661.700012      2.163691         0.100000
898.000000      2.360335         0.100000
1173.199951     2.469585         0.100000
1332.500000     2.610782         0.100000
1836.099976     2.728864         0.100000
```

```
#Efficiency
88.000000         0.077359         0.003270
122.099998        0.084491         0.002780
158.899994        0.078352         0.002198
320.100006        0.045631         0.001345
391.700012        0.039058         0.001063
514.000000        0.031686         0.000824
661.700012        0.026765         0.000931
898.000000        0.018370         0.000472
1173.199951       0.015078         0.000525
1332.500000       0.013392         0.000466
1836.099976       0.010400         0.000269
#Spectrum
4096  0
0     0           0          0          0          0
5     0           0          0          0          0
10    0           0          0          0          0
15    0           0          0          0          0
20    0           0          0          0          0
25    0           0          0          0          0
30    0           0          0          0          0
35    0           0          0          0          0
40    0           0          0          0          0
45    0           0          0          0          0
50    0           0          0          0          0
55    161         1218       1038       872        791
60    763         711        720        711        706
65    676         681        704        720        662
70    700         661        708        681        697
75    671         710        636        631        712
80    689         694        655        683        638
85    658         653        659        604        639
90    617         686        697        696        672
95    662         622        635        594        643
100   670         624        561        636        637
105   625         636        627        632        651
110   657         600        673        639        680
```

… 5 channels per line to the end…

```
4080  10          17         16         19         17
4085  16          12         11         21         15
4090  8           14         24         17         10
4095  12          0          0          0          0
STOP
```

Appendix B - Reporting format

The format for this information is free but a structured and clear format is preferred. Here is an example on a typical result file containing all necessary information:

```
Sample description      : CA002 CA002CAA2 P DISK FULL 0
Spectrum file           : 36217

Start of sampling       :1998-Jul-01    at 02:28:59
End of sampling         :1998-Jul-02    at 02:01:30
Acquisition started at  :1998-Jul-03    at 02:20:32
Acquisition ended at    :1998-Jul-04    at 01:34:15

Dead time               :       0.01 %
Live time               :      23.23 h
Real time               :      23.23 h
Decay time              :      24.32 h
Sampling time           :      23.54 h

Air quantity            :   22606.00 m3 (960.24 m3/h)

MDA decision limit 1-alpha:   95.00 %
MDA detection limit 1-beta:   95.00 %

***************   P E A K    R E P O R T   *****************
     energy    area    error background intensity
        keV    counts      %     counts       gps

      74.38 6.43E+003    1.4 1.50E+003 6.99E-001
      76.75 9.56E+003    1.1 1.38E+003 1.04E+000
      86.83 3.48E+003    2.0 1.20E+003 3.71E-001
      89.55 9.96E+002    5.1 1.17E+003 1.06E-001
     238.12 2.31E+004    0.7 7.61E+002 3.01E+000
     276.85 9.54E+002    4.4 5.63E+002 1.30E-001
     299.55 1.17E+003    3.8 5.78E+002 1.63E-001
     477.11 7.31E+003    1.2 2.77E+002 1.92E+000
     510.30 2.97E+003    2.0 3.19E+002 8.22E-001
     582.67 7.24E+003    1.2 2.05E+002 2.24E+000
     726.75 1.50E+003    2.9 1.73E+002 5.75E-001
     859.97 8.94E+002    3.8 1.54E+002 4.10E-001
    1460.38 2.97E+003    1.9 7.78E+001 2.10E+000
```

```
********  R A D I O N U C L I D E    R E P O R T  *********

     Nuclide     Activity    Activity  Error
           Bq/sample      µBq/m3      %

     Be-7           18.82       832.3   1.23
     K-40           19.71         872   1.89
     Tl-208Th       49.89        2207   1.13
     Pb-212Th       126.7        5603   0.67
     Bi-212Th       161.5        7145   2.85

*****  D E T E C T I O N   L I M I T   R E P O R T  *****

     Nuclide       MDA [Bq]      MDC [µBq/m3]

     Cr-51           0.182        8.036
     Mn-54           0.031        1.364
     Co-57           0.019        0.853
     Co-58           0.031        1.375
     Co-60           0.032        1.430
     Zn-65           0.071        3.150
     Zr-95           0.049        2.167
     Zr-97           0.309       13.667
     Nb-95           0.028        1.235
     Mo-99           0.037        1.645
     Tc-99m          8.160      360.958
     Ru-97           0.035        1.570
     Ru-103          0.025        1.120
     Te-132          0.035        1.545
     I-131           0.031        1.354
     I-132       14606.096   646115.875
     I-133           0.180        7.978
     I-134     4.9938e+010 2.20906e+012
     I-135          31.223     1381.162
     Cs-134          0.030        1.345
     Cs-136          0.032        1.394
     Cs-137          0.031        1.388
     Ba-133          0.028        1.244
     Ba-140          0.120        5.288
     La-140          0.093        4.134
     Ce-141          0.040        1.771
     Ce-143          0.147        6.508
     Ce-144          0.180        7.949
     Nd-147          0.078        3.466
```

Appendix C - Detection limits

The participants are asked to report the detection limits using the method commonly in use or that would be used in such a situation. One method often referred to is the one by L.A. Currie ("Limits for Quantitative Detection and Quantitative Determination, Analytical Chemistry", 40 (3), 586-593 (1968), 95% probability level). The detection limits are requested for following isotopes:

> Zr-95, Nb-95, Zr-97, Mo-99, Tc-99m, Ru-103, I-131, I-132, I-133, I-134, I-135, Te-132, Cs-134, Cs-137, Ba-140, La-140, Ce-141, Ce-143, Ce-144, Nd-147

As a rough rule the detection limit for a peak area at the 95% probability level (Currie) can be calculated as follows:

$$L_D = k^2 + 2L_C = 2.71 + 4.65\sqrt{B}\,.$$

Where B is the number of counts at the baseline where the peak should be. There are a number of definitions how to calculate the B itself and therefore the laboratories should also explain how wide a region has been selected (preferably in FWHM’s) for this calculation. Optimum value for this is +- 0.71 FWHM.

Appendix D - Time schedules for the exercises

The first exercise, October - November 2000

- September: Introduction of the exercise, test spectra made available for downloading, assistance to participants if needed.
- September: Registration of participants in exercise.
- October 2: Last day of registration
- October 4: Exercise spectra sent out at 9:00 UTC (11:00 Scandinavian + Estonian time; 12:00 Finnish + Latvian + Lithuanian time)
  A link to the same information was also opened at the same time on this web page for those who prefer downloading from the Web.
- October 4: Quick analysis results were required from the laboratories
- October 11: A preliminary report available through BOK 1.1 web site
- November 3 Comprehensive analysis results required from the laboratories

The second exercise, May - June 2001

- April: Introduction of the exercise, test spectra made available for downloading, assistance to participants provided as requested.
- April 30: File conversion utility available for participants from the web site.
- May: Registration of participants in exercise.
- May 12: Last day of registration
- May 16: Exercise spectra sent out at 9:00 UTC (11:00 Scandinavian + Estonian + Lithuanian time; 12:00 Finnish time + Latvian time).
- A link to the same information was also opened at exactly the same time on the BOK-1.1 web page for those who prefer downloading from the Web.
- May 16: Quick analysis results required from the laboratories
- June 4. Deadline for comprehensive analysis results from the laboratories

Appendix E - Added peaks and their areas in the second exercise

Altogether 246 peaks were added, one peak (76) had negative area and it was ignored during the synthesis process.

```
I    Channel    Energy    Energy    Peak Area Intensity Intensity
KeV       error KeV counts    GPS       error

  1     58.623      48.103        0.164   1.923E+004    5.305e+006       32.208
  2     61.688      49.614        0.100   4.387E+005    1.037e+008       26.564
  3     81.790      59.521        0.116   9.267E+003    1.039e+006       17.844
  4     85.422      61.311        0.116   9.382E+003    9.543e+005       16.844
  5     96.559      66.800        0.101   6.662E+004    5.299e+006        9.459
  6    112.733      74.771        0.115   2.816E+004    1.743e+006        8.199
  7    117.190      76.968        0.110   3.321E+004    1.950e+006        8.550
  8    123.356      80.007        0.103   2.616E+005    1.443e+007        4.830
  9    124.955      80.795        0.126   8.447E+004    4.591e+006       13.608
 10    136.287      86.380        0.105   5.813E+004    2.896e+006        5.138
 11    145.499      90.920        0.106   6.597E+004    3.119e+006        5.267
 12    148.763      92.529        0.260   1.109E+004    5.165e+005       25.473
 13    158.000      97.081        0.100   1.940E+004    8.728e+005       52.698
 14    162.793      99.444        0.110   2.068E+005    9.175e+006        5.883
 15    167.449     101.738        0.184   6.814E+004    2.989e+006       13.559
 16    171.270     103.622        0.102   3.309E+005    1.441e+007        4.190
 17    176.049     105.977        0.102   4.059E+005    1.754e+007        4.552
 18    187.796     111.767        0.101   3.242E+005    1.386e+007        3.809
 19    197.186     116.394        0.100   4.469E+005    1.907e+007        3.613
 20    205.837     120.658        0.149   4.290E+004    1.834e+006       13.153
 21    208.696     122.067        0.229   2.451E+004    1.050e+006       21.996
 22    227.306     131.239        0.166   1.278E+004    5.573e+005       17.171
 23    231.720     133.415        0.104   5.914E+004    2.593e+006        4.925
 24    237.347     136.188        0.127   1.502E+004    6.640e+005        9.807
 25    242.314     138.636        0.118   2.387E+004    1.064e+006        9.633
 26    245.907     140.406        0.100   2.209E+006    9.901e+007        3.709
 27    255.939     145.351        0.101   3.695E+005    1.687e+007        4.154
 28    259.453     147.083        0.163   3.048E+004    1.401e+006       21.019
 29    264.646     149.642        0.112   8.139E+004    3.780e+006        8.494
 30    271.835     153.185        0.107   7.674E+004    3.617e+006        8.364
 31    290.911     162.587        0.101   1.943E+005    9.555e+006        4.319
 32    293.418     163.822        0.126   4.410E+004    2.181e+006        6.036
 33    298.191     166.175        0.474   8.560E+003    4.280e+005       13.627
 34    319.407     176.631        0.100   1.587E+005    8.345e+006        4.803
 35    328.341     181.034        0.101   1.255E+005    6.743e+006        4.987
 36    332.825     183.244        0.117   2.752E+004    1.495e+006        8.339
 37    338.297     185.941        0.125   2.469E+004    1.359e+006        9.118
 38    353.622     193.494        0.316   1.423E+004    8.133e+005       16.172
 39    358.611     195.953        1.174   8.568E+003    4.958e+005       56.153
 40    367.904     200.533        0.279   2.280E+004    1.349e+006        8.715
 41    383.343     208.142        0.136   2.146E+004    1.319e+006       10.405
 42    386.303     209.601        0.104   5.742E+004    3.553e+006        5.939
 43    419.156     225.793        0.364   9.655E+004    6.458e+006       50.713
 44    423.754     228.059        0.104   8.612E+006    5.822e+008        7.807
 45    425.201     228.772        0.205   2.054E+006    1.393e+008       22.730
 46    445.062     238.560        0.100   1.596E+005    1.132e+007        5.920
```

47	449.542	240.769	0.132	3.330E+004	2.386e+006	13.395
48	451.610	241.788	0.143	2.505E+004	1.804e+006	17.431
49	458.840	245.351	0.166	9.438E+003	6.904e+005	22.156
50	469.095	250.405	0.127	8.989E+003	6.724e+005	12.860
51	473.612	252.631	0.103	3.174E+004	2.398e+006	6.677
52	478.089	254.838	0.102	3.255E+004	2.482e+006	6.626
53	494.222	262.789	0.101	1.627E+005	1.284e+007	6.054
54	501.856	266.551	0.147	6.952E+003	5.571e+005	12.699
55	509.222	270.182	0.151	8.392E+003	6.825e+005	10.429
56	516.139	273.591	0.101	7.651E+004	6.310e+006	6.280
57	524.171	277.550	0.102	1.528E+005	1.280e+007	6.205
58	537.922	284.327	0.100	5.567E+005	4.791e+007	6.212
59	556.085	293.279	1.144	9.703E+003	8.640e+005	61.530
60	559.829	295.124	0.105	3.922E+004	3.516e+006	7.476
61	570.002	300.137	0.693	8.668E+003	7.918e+005	22.394
62	579.505	304.821	0.202	7.699E+004	7.153e+006	9.412
63	583.162	306.624	0.832	9.376E+003	8.768e+005	12.858
64	589.898	309.943	0.889	9.698E+003	9.178e+005	15.437
65	602.142	315.978	0.112	2.195E+004	2.122e+006	7.928
66	605.330	317.549	0.101	6.323E+003	6.144e+005	17.723
67	609.020	319.367	0.155	1.030E+004	1.007e+006	11.471
68	621.981	325.755	0.116	2.066E+004	2.064e+006	9.927
69	627.941	328.693	0.107	3.541E+005	3.572e+007	6.702
70	639.188	334.236	0.154	2.870E+004	2.949e+006	9.880
71	643.221	336.224	0.271	2.580E+004	2.667e+006	10.221
72	647.421	338.293	0.118	3.193E+004	3.324e+006	10.224
73	651.987	340.544	0.101	2.326E+005	2.438e+007	6.758
74	655.266	342.159	0.112	6.064E+004	6.389e+006	7.777
75	674.942	351.855	0.168	6.353E+004	6.896e+006	8.230
76	684.789	356.707	0.138	-9.57E+003	-1.05e+006	37.508
77	697.137	362.792	0.127	7.151E+004	8.019e+006	10.691
78	700.436	364.418	0.100	5.026E+006	5.663e+008	6.831
79	747.714	387.714	0.103	2.616E+004	3.145e+006	7.419
80	796.718	411.861	0.247	6.400E+003	8.182e+005	13.465
81	802.646	414.782	0.304	6.826E+003	8.790e+005	17.610
82	806.502	416.682	0.176	3.473E+004	4.492e+006	7.864
83	820.674	423.665	0.112	4.053E+004	5.331e+006	7.723
84	829.069	427.802	0.167	1.553E+004	2.063e+006	9.050
85	838.012	432.209	0.103	6.617E+004	8.879e+006	8.180
86	849.031	437.638	0.174	2.496E+004	3.391e+006	9.448
87	862.619	444.334	0.554	1.151E+004	1.587e+006	14.540
88	866.160	446.078	0.126	3.765E+004	5.211e+006	10.464
89	878.812	452.313	0.123	7.629E+003	1.071e+006	11.296
90	893.357	459.480	0.108	7.229E+004	1.030e+007	7.421
91	900.538	463.018	0.102	2.007E+004	2.882e+006	7.661
92	921.165	473.182	0.105	6.750E+004	9.894e+006	8.836
93	925.466	475.302	0.160	2.277E+004	3.353e+006	14.970
94	931.519	478.284	0.175	1.712E+004	2.535e+006	17.550
95	949.187	486.990	0.100	5.255E+005	7.917e+007	7.576
96	951.529	488.144	0.135	2.866E+004	4.328e+006	15.299
97	969.582	497.040	0.100	1.028E+006	1.580e+008	7.583
98	981.910	503.115	0.103	2.699E+004	4.192e+006	8.230
99	987.377	505.808	0.100	2.905E+005	4.536e+007	7.633
100	997.080	510.590	0.112	3.073E+004	4.840e+006	9.286
101	999.544	511.804	0.103	6.426E+004	1.014e+007	8.098
102	1021.532	522.638	0.100	9.419E+005	1.516e+008	7.699
103	1036.181	529.857	0.101	1.126E+005	1.835e+007	7.892
104	1047.380	535.375	0.118	2.659E+004	4.375e+006	9.914
105	1051.193	537.254	0.102	2.518E+005	4.156e+007	7.799
106	1070.847	546.939	0.152	6.399E+004	1.074e+007	7.857

107	1091.185	556.960	0.119	7.255E+003	1.238e+006	10.973
108	1103.848	563.200	0.101	7.944E+004	1.369e+007	8.066
109	1116.227	569.300	0.100	1.452E+005	2.527e+007	7.958
110	1125.511	573.875	0.135	1.220E+004	2.138e+006	12.537
111	1144.443	583.204	0.100	7.274E+004	1.293e+007	7.980
112	1160.753	591.240	0.136	4.932E+003	8.879e+005	12.722
113	1179.496	600.476	0.192	1.449E+004	2.645e+006	14.410
114	1188.023	604.678	0.100	8.826E+005	1.621e+008	8.029
115	1197.392	609.294	0.125	5.826E+004	1.077e+007	14.543
116	1199.349	610.259	0.124	5.469E+004	1.013e+007	15.361
117	1221.081	620.967	0.105	1.003E+005	1.885e+007	10.539
118	1222.886	621.857	0.171	2.475E+004	4.660e+006	28.199
119	1239.813	630.197	0.100	6.759E+005	1.288e+008	8.106
120	1253.566	636.974	0.100	2.480E+005	4.770e+007	8.149
121	1265.313	642.763	0.175	5.092E+003	9.871e+005	16.290
122	1281.130	650.556	0.100	1.258E+005	2.464e+007	8.182
123	1296.198	657.981	0.292	2.074E+004	4.105e+006	21.989
124	1303.625	661.641	0.107	1.222E+006	2.431e+008	8.257
125	1315.901	667.686	0.104	4.611E+006	9.244e+008	8.230
126	1320.167	669.787	0.155	2.307E+005	4.637e+007	9.293
127	1323.740	671.547	0.158	1.921E+005	3.870e+007	11.032
128	1350.865	684.904	0.100	5.800E+004	1.189e+007	8.309
129	1372.920	695.765	0.101	3.489E+004	7.255e+006	8.382
130	1376.357	697.458	0.116	7.233E+003	1.507e+006	10.152
131	1395.071	706.674	0.126	5.080E+003	1.071e+006	10.703
132	1424.014	720.927	0.161	7.254E+003	1.556e+006	13.046
133	1427.980	722.880	0.105	6.053E+004	1.302e+007	8.830
134	1430.710	724.224	0.103	7.165E+004	1.543e+007	8.801
135	1436.320	726.987	0.100	2.537E+005	5.482e+007	8.414
136	1440.089	728.843	0.101	5.931E+004	1.285e+007	8.543
137	1461.617	739.444	0.100	6.535E+004	1.434e+007	8.463
138	1486.350	751.624	0.100	3.276E+004	7.294e+006	8.624
139	1496.589	756.666	0.102	8.520E+004	1.908e+007	8.500
140	1515.087	765.776	0.100	1.463E+005	3.312e+007	8.522
141	1528.986	772.620	0.100	3.078E+006	7.024e+008	8.545
142	1540.126	778.106	0.208	9.353E+003	2.148e+006	28.731
143	1544.494	780.257	0.122	3.571E+004	8.223e+006	11.221
144	1549.240	782.595	0.146	2.169E+004	5.009e+006	14.844
145	1554.312	785.092	0.191	1.231E+004	2.851e+006	11.843
146	1567.457	791.565	0.117	4.983E+003	1.163e+006	13.704
147	1572.368	793.984	0.112	2.112E+004	4.940e+006	9.818
148	1576.070	795.807	0.100	5.941E+005	1.393e+008	8.604
149	1588.448	801.903	0.101	5.886E+004	1.389e+007	8.694
150	1604.488	809.801	0.101	1.045E+005	2.488e+007	8.663
151	1609.490	812.264	0.102	2.233E+005	5.334e+007	8.655
152	1616.574	815.753	0.100	1.677E+005	4.020e+007	8.669
153	1622.077	818.463	0.100	2.152E+005	5.177e+007	8.673
154	1630.711	822.715	0.114	5.562E+003	1.344e+006	11.367
155	1656.746	835.536	0.727	3.501E+003	8.585e+005	10.082
156	1665.928	840.058	0.151	1.862E+003	4.589e+005	18.590
157	1690.538	852.177	0.101	1.585E+004	3.959e+006	8.899
158	1698.893	856.291	0.102	1.538E+003	3.858e+005	12.278
159	1707.488	860.524	0.104	9.436E+003	2.379e+006	9.130
160	1712.950	863.214	0.101	2.057E+004	5.200e+006	8.868
161	1722.202	867.770	0.101	3.752E+004	9.535e+006	8.854
162	1738.028	875.563	0.128	1.336E+004	3.423e+006	11.333
163	1741.098	877.075	0.110	3.100E+004	7.958e+006	9.411
164	1756.546	884.682	0.198	1.155E+004	2.990e+006	10.736
165	1765.714	889.197	0.544	1.499E+003	3.899e+005	29.666
166	1796.823	904.517	0.130	2.215E+003	5.858e+005	14.317

167	1808.177	910.108	0.107	2.721E+004	7.241e+006	10.786
168	1810.186	911.098	0.101	6.625E+004	1.765e+007	9.273
169	1819.094	915.484	0.191	1.582E+003	4.233e+005	80.245
170	1827.337	919.544	0.395	1.791E+004	4.814e+006	27.973
171	1838.699	925.139	0.168	4.354E+004	1.177e+007	14.652
172	1843.842	927.671	0.867	1.311E+004	3.556e+006	33.405
173	1857.071	934.186	0.110	3.293E+003	8.993e+005	11.512
174	1863.785	937.492	0.110	4.937E+003	1.353e+006	9.528
175	1889.762	950.285	0.188	6.235E+003	1.732e+006	13.675
176	1894.544	952.640	0.236	9.594E+003	2.672e+006	15.525
177	1898.513	954.595	0.101	5.633E+005	1.572e+008	9.091
178	1919.267	964.815	0.101	1.106E+004	3.121e+006	9.244
179	1927.677	968.956	0.105	3.220E+004	9.126e+006	9.147
180	1958.994	984.378	0.101	2.212E+004	6.375e+006	9.262
181	2062.118	1035.162	0.102	1.654E+004	5.031e+006	9.606
182	2069.052	1038.576	0.116	4.666E+003	1.424e+006	11.533
183	2088.336	1048.073	0.100	1.377E+005	4.247e+007	9.391
184	2161.800	1084.230	0.636	4.962E+003	1.590e+006	15.380
185	2166.597	1086.591	1.207	4.183E+003	1.344e+006	11.344
186	2235.048	1120.281	0.103	1.644E+004	5.475e+006	9.851
187	2245.684	1125.516	0.114	7.345E+003	2.461e+006	10.781
188	2249.724	1127.505	0.136	1.891E+003	6.347e+005	20.813
189	2267.091	1136.052	0.100	8.541E+004	2.894e+007	9.664
190	2282.479	1143.626	0.103	3.933E+004	1.344e+007	9.781
191	2291.951	1148.288	0.132	8.838E+003	3.034e+006	11.024
192	2305.040	1154.730	0.236	1.981E+003	6.851e+005	14.653
193	2331.927	1167.963	0.109	9.495E+003	3.330e+006	9.777
194	2342.607	1173.220	0.101	4.006E+004	1.413e+007	9.769
195	2377.099	1190.196	0.110	4.649E+003	1.671e+006	10.879
196	2410.578	1206.674	0.112	5.905E+003	2.160e+006	10.563
197	2468.816	1235.338	0.193	2.926E+004	1.105e+007	12.729
198	2474.433	1238.102	1.346	6.647E+003	2.518e+006	15.443
199	2507.468	1254.361	0.137	2.226E+003	8.582e+005	13.354
200	2544.792	1272.731	0.118	4.444E+003	1.749e+006	11.209
201	2548.605	1274.608	0.127	3.433E+003	1.354e+006	11.972
202	2581.324	1290.712	0.201	2.764E+004	1.110e+007	10.067
203	2590.657	1295.305	0.191	4.264E+004	1.721e+007	10.239
204	2595.992	1297.931	0.246	1.990E+004	8.055e+006	10.135
205	2629.912	1314.626	0.149	1.871E+003	7.718e+005	12.341
206	2636.356	1317.797	0.114	2.940E+003	1.217e+006	11.097
207	2666.251	1332.511	0.104	6.155E+003	2.591e+006	10.545
208	2732.611	1365.172	0.126	1.383E+004	6.042e+006	10.266
209	2746.489	1372.002	0.101	5.686E+004	2.504e+007	10.085
210	2757.892	1377.615	0.208	3.690E+003	1.635e+006	13.210
211	2772.065	1384.590	0.215	2.332E+003	1.042e+006	12.372
212	2800.452	1398.562	0.100	1.567E+005	7.116e+007	10.059
213	2803.883	1400.250	0.167	6.757E+003	3.074e+006	17.065
214	2820.354	1408.357	0.261	2.408E+003	1.106e+006	14.844
215	2826.713	1411.487	0.351	8.851E+002	4.080e+005	27.048
216	2885.002	1440.175	0.101	1.439E+004	6.862e+006	10.081
217	2889.702	1442.489	0.100	3.277E+004	1.566e+007	9.995
218	2917.736	1456.286	0.227	3.052E+003	1.483e+006	17.094
219	2922.854	1458.806	0.337	7.953E+003	3.875e+006	28.009
220	2926.976	1460.834	0.102	3.094E+005	1.511e+008	9.966
221	2958.847	1476.520	0.111	3.195E+003	1.590e+006	10.655
222	2964.954	1479.526	0.195	8.057E+002	4.024e+005	16.878
223	2998.521	1496.047	0.191	9.480E+002	4.829e+005	16.281
224	3006.345	1499.898	0.148	1.588E+003	8.127e+005	12.275
225	3016.271	1504.784	0.170	9.735E+002	5.011e+005	15.233
226	3025.381	1509.267	0.123	1.745E+003	9.031e+005	12.083

227	3045.913	1519.372	0.130	1.505E+003	7.886e+005	12.070
228	3186.072	1588.356	0.113	5.878E+003	3.350e+006	10.401
229	3195.183	1592.840	0.123	1.294E+004	7.415e+006	9.893
230	3202.002	1596.196	0.101	3.571E+005	2.055e+008	9.289
231	3252.614	1621.106	0.124	4.564E+003	2.709e+006	11.263
232	3256.444	1622.992	1.022	8.109E+002	4.824e+005	22.386
233	3272.028	1630.662	0.123	2.771E+003	1.664e+006	9.477
234	3287.541	1638.297	0.189	8.577E+002	5.202e+005	15.196
235	3334.358	1661.339	0.132	1.209E+003	7.548e+005	11.487
236	3468.259	1727.243	0.115	3.338E+003	2.267e+006	8.367
237	3473.411	1729.778	0.115	2.804E+003	1.911e+006	8.713
238	3529.085	1757.180	0.108	4.291E+003	3.030e+006	7.505
239	3535.743	1760.457	0.137	1.325E+003	9.396e+005	9.173
240	3544.225	1764.631	0.101	1.478E+004	1.054e+007	7.047
241	3572.720	1778.656	0.125	1.128E+003	8.191e+005	10.016
242	3712.711	1847.557	0.111	1.871E+003	1.489e+006	6.303
243	3794.542	1887.832	0.156	6.871E+002	5.773e+005	10.963
244	3862.167	1921.116	0.100	2.038E+004	1.792e+007	3.862
245	4027.149	2002.317	0.101	1.986E+004	1.954e+007	7.796
246	4043.708	2010.467	0.143	8.820E+002	8.776e+005	8.414

Appendix F - C-programming language files

MAIN.C

```
/*
SPECONV spectrum file converter.
*/

#include "all.h"
FILE  *asc_stream;
runrecord run;

int main(int argc,char *argv[])
    {
    sprintf(run.version,"CONVSPE spectrum file converter. Version 1.05");
    printf("%s\n",run.version);
               specta_1.iset=0;

    if(argc!=5)
        {
        printf("usage: convspe -type inputfile -type outpufile\n\n");
        printf("type may be:\n");
        printf("       -rms (RMS 2.0 file format)\n");
        printf("       -asc (Plain ASCII file, one channel on each line)\n");
        printf("       -chn (Ortec CHN file format)\n");
        printf("       -spc (Ortec SPC file format)\n\n");
        printf("email: doletum@kolumbus.fi for more information.\n");
        exit(0);
        }

      specta_1.isoiy=2000;
      specta_1.isoim=1;
      specta_1.isoid=1;
      specta_1.isoih=0;
      specta_1.isoimi=0;
      specta_1.isois=0;
      specta_1.ieoiy=2000;
      specta_1.ieoim=1;
      specta_1.ieoid=1;
      specta_1.ieoih=0;
      specta_1.ieoimi=0;
      specta_1.ieois=0;
      specta_1.isocy=2000;
      specta_1.isocm=1;
      specta_1.isocd=1;
      specta_1.isoch=0;
      specta_1.isocmi=0;
      specta_1.isocs=0;
      specta_1.ieocy=2000;
      specta_1.ieocm=1;
      specta_1.ieocd=1;
      specta_1.ieoch=0;
      specta_1.ieocmi=0;
      specta_1.ieocs=0;
      specta_1.svol=1.0;

    if(strcmp(argv[1],"-spc")==0 || strcmp(argv[3],"-spc")==0)
    if(!spc_read_spectrum("convspe.bin",0))
            {
            printf("ERROR: SPC Init file convspe.bin was not found from current directory.\n");
            exit(0);
            }

    if(strcmp(argv[1],"-rms")==0)
        {
                if(!rms_read_spectrum(argv[2],0))
                    {
                    printf("RMS Spectrum %s was not found\n",argv[2]);
                    exit(0);
                    }
        else
            rms_read_calibr(argv[2],0);
            }
```

```
if(strcmp(argv[1],"-chn")==0)
    {
             if(!chn_read_spectrum(argv[2],0))
                {
                printf("CHN Spectrum %s was not found\n",argv[2]);
                exit(0);
                }
             }

if(strcmp(argv[1],"-asc")==0)
    {
             if(!asc_read_spectrum(argv[2],0))
                {
                printf("ASCII Spectrum %s was not found\n",argv[2]);
                exit(0);
                }
             }

if(strcmp(argv[1],"-spc")==0)
    {
        if(!spc_read_spectrum(argv[2],0))
                {
                printf("SPC Spectrum %s was not found\n",argv[2]);
                exit(0);
                }
             }

if(specta_1.iset==0)
                {
                printf("Spectrum was not read in.\n");
                exit(0);
                }


if(strcmp(argv[3],"-rms")==0)
    {
             if(!rms_write_spectrum(argv[4],0))
                 {
                 printf("RMS Spectrum %s was not written.\n",argv[4]);
                 exit(0);
                 }
    exit(1);
         }

if(strcmp(argv[3],"-chn")==0)
    {
             if(!chn_write_spectrum(argv[4],0))
                {
                printf("CHN Spectrum %s was not written.\n",argv[4]);
                exit(0);
                }
    exit(1);
             }

if(strcmp(argv[3],"-asc")==0)
    {
             if(!asc_write_spectrum(argv[4],0))
                {
                printf("ASCII Spectrum %s was not written.\n",argv[4]);
                exit(0);
                }
    exit(1);
             }

if(strcmp(argv[3],"-spc")==0)
    {
             if(!spc_write_spectrum(argv[4],0))
                {
                printf("SPC Spectrum %s was not written.\n",argv[4]);
                exit(0);
                }
    exit(1);
             }
    printf("Unknown spectrum type: %s.\n",argv[3]);
        exit(0);
    }

void error_out(char *c)
    {
    printf("Error: %s\n",c);
    }
```

TIME.C

```
/*
CONVSPE time and date handling
*/

#include "all.h"

int i1;
char c1[256];
char date_str[256];
char time_str[256];
int european_dst=1;

char months[12][4]={"Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct","Nov","Dec"};
int month_lenghts[12]={31,28,31,30,31,30,31,31,30,31,30,31};

#define D_400 146097L
#define D_100 36524L
#define D_4 1461L
#define D_1 365L
#define M_DAY 86400L
#define M_HOUR 3600L
#define M_MIN 60L



void dttoi(int *j1,int *j2,int *j3,int *j4,int *j5,int *j6,char *str,int len)
   {
   char prev_char;
   int i,j,k;
   int spaces;
   strncpy(c1,str,len);
   c1[len]=0;
   i=0;
   j=0;
   k=0;
   date_str[30]=0;
   time_str[30]=0;
   while(c1[i]==' ')
      {
      i++;
      k++;
      }
   spaces=0;
   prev_char=0;
   while( c1[i]!=',' && spaces<3  &&  i<30  &&  i<len && c1[i]!=0)
      {
      if(prev_char!=' ' && prev_char!='-'&& prev_char!='/') if(c1[i]==' ') spaces++;
      if(c1[i]=='-') spaces++;
      if(c1[i]=='/') spaces++;
      date_str[i-k]=c1[i];
      prev_char=c1[i];
      i++;
      }
   date_str[i]=0;

   if(c1[i]==',' || spaces<3)i++;

   k=0;
   while(c1[j+i]==' ')
      {
      k++;
      j++;
      }
```

```
spaces=0;
   prev_char=0;
   while( spaces<3 && j<30 && (j+i)<len && c1[i]!=0)
      {
      if(prev_char!=' ' && prev_char!=':')if(c1[j+i]==' ') spaces++;
      if(c1[j+i]==':') spaces++;
      time_str[j-k]=c1[j+i];
      prev_char=c1[j+i];
      j++;
      }
   time_str[j]=0;

   dattoi(j1,j2,j3,date_str,30);
   timtoi(j4,j5,j6,time_str,30);
   }

void itodt(int *j1,int *j2,int *j3,int *j4,int *j5,int *j6,char *str,int len)
   {
   char hstr[6];
   char mstr[6];
   char sstr[6];
   char ystr[6];
   char mostr[6];
   char dstr[6];
   sprintf(ystr,"%d",*j1);
   sprintf(dstr,"%d",*j3);
   sprintf(hstr,"%d",*j4);
   sprintf(mstr,"%d",*j5);
   sprintf(sstr,"%d",*j6);
   sprintf(mostr,"%s",months[*j2-1]);
   if(*j1<10)sprintf(ystr,"0%d",*j1);
   if(*j3<10)sprintf(dstr,"0%d",*j3);
   if(*j4<10)sprintf(hstr,"0%d",*j4);
   if(*j5<10)sprintf(mstr,"0%d",*j5);
   if(*j6<10)sprintf(sstr,"0%d",*j6);
   sprintf(c1,"%s-%s-%s,%s:%s:%s             ",ystr,mostr,dstr,hstr,mstr,sstr);
   c1[len]=0;
   strncpy(str,c1,len);
   str[len]='\0';
   }

void itodt_short(int *j1,int *j2,int *j3,int *j4,int *j5,int *j6,char *str,int len)
   {
   char hstr[6];
   char mstr[6];
   char sstr[6];
   char ystr[6];
   char mostr[6];
   char dstr[6];
   sprintf(ystr,"%d",*j1);
   sprintf(mostr,"%d",*j2);
   sprintf(dstr,"%d",*j3);
   sprintf(hstr,"%d",*j4);
   sprintf(mstr,"%d",*j5);
   sprintf(sstr,"%d",*j6);
   if(*j1<10)sprintf(ystr,"0%d",*j1);
   if(*j2<10)sprintf(mostr,"0%d",*j2);
   if(*j3<10)sprintf(dstr,"0%d",*j3);
   if(*j4<10)sprintf(hstr,"0%d",*j4);
   if(*j5<10)sprintf(mstr,"0%d",*j5);
   if(*j6<10)sprintf(sstr,"0%d",*j6);
   sprintf(c1,"%s%s%s %s%s%s             ",ystr,mostr,dstr,hstr,mstr,sstr);
   c1[len]=0;
   strncpy(str,c1,len);
   }

void dattoi(int *j1,int *j2,int *j3,char *str,int len)
   {
   char year[6],month[6],day[6];
   int i,j,k;
   strncpy(c1,str,len);
   c1[len]=0;
   i=0;
   j=0;
   k=0;
   year[5]=0;
   month[5]=0;
   day[5]=0;
   while( c1[i]!='-' && c1[i]!='/' && c1[i]!= ' ' &&  i<5 && i<len)
      {
      year[i]=c1[i];
      i++;
      }
   year[i]=0;
   i++;
   while( c1[j+i]!='-' && c1[j+i]!='/' && c1[j+i]!=' ' && j<5 && (j+i)<len)
      {
      month[j]=c1[j+i];
      j++;
      }
   month[j]=0;
   j+=i;
   j++;
   while( c1[k+j]!=' ' && k<5 && (k+j)<len)
      {
      day[k]=c1[k+j];
      k++;
      }
   day[k]=0;
   *j1=atoi(year);
   if(!isdigit(month[0]))
      {
      month[3]=0;
      for(i=0;i<12;i++)
         if(strcmp(month,months[i])==0)*j2=i+1;
      }
   else
     *j2=atoi(month);
   *j3=atoi(day);
   }

void itodat(int *j1,int *j2,int *j3,char *str,int len)
   {
   char ystr[6];
   char mstr[6];
   char dstr[6];
   sprintf(ystr,"%d",*j1);
   sprintf(dstr,"%d",*j3);
   if(*j1<10)sprintf(ystr,"0%d",*j1);
   if(*j3<10)sprintf(dstr,"0%d",*j3);
   sprintf(mstr,"%s",months[*j2-1]);
   sprintf(c1,"%s-%s-%s             ",ystr,mstr,dstr);
   strncpy(str,c1,len+1);
   str[len]=0;
   }

void itodatortec(int *j1,int *j2,int *j3,char *str,int len)
   {
   char ystr[6];
   char mstr[6];
   char dstr[6];
   sprintf(ystr,"%d",*j1);
   sprintf(dstr,"%d",*j3);
   if(*j1<2000)
      {
      *j1-=1900;
      if(*j1<10)
         sprintf(ystr,"0%d0",*j1);
      else
         sprintf(ystr,"%d0",*j1);
      }
   else
      {
      *j1-=2000;
      if(*j1<10)
         sprintf(ystr,"0%d1",*j1);
      else
         sprintf(ystr,"%d1",*j1);
      }
   if(*j3<10)sprintf(dstr,"0%d",*j3);
   sprintf(mstr,"%s",months[*j2-1]);
   sprintf(c1,"%s-%s-%s             ",dstr,mstr,ystr);
   strncpy(str,c1,len+1);
   str[len]=0;
   }

void itotim(int *j1,int *j2,int *j3,char *str,int len)
   {
   char hstr[6];
   char mstr[6];
   char sstr[6];
   sprintf(hstr,"%d",*j1);
   sprintf(mstr,"%d",*j2);
   sprintf(sstr,"%d",*j3);
   if(*j1<10)sprintf(hstr,"0%d",*j1);
   if(*j2<10)sprintf(mstr,"0%d",*j2);
   if(*j3<10)sprintf(sstr,"0%d",*j3);
   sprintf(c1,"%s:%s:%s             ",hstr,mstr,sstr);
   strncpy(str,c1,len);
   str[len]=0;
   }

void timtoi(int *j1,int *j2,int *j3,char *str,int len)
   {
   char hour[6],min[6],sec[6];
   int i,j,k;
   strncpy(c1,str,len);
   c1[len]=0;
   i=0;
   j=0;
   k=0;
   hour[5]=0;
   min[5]=0;
   sec[5]=0;
   while( c1[i]!=':' && c1[i]!= ' ' &&  i<5 && i<len)
      {
      hour[i]=c1[i];
      i++;
      }
   hour[i]=0;
   i++;
   while( c1[j+i]!=':' && c1[j+i]!=' ' && j<5 && (j+i)<len)
      {
      min[j]=c1[j+i];
      j++;
      }
   min[j]=0;
   j+=i;
   j++;
   while( c1[k+j]!=' ' && k<5 && (k+j)<len)
      {
      sec[k]=c1[k+j];
      k++;
      }
   sec[k]=0;
   *j1=atoi(hour);
   *j2=atoi(min);
   *j3=atoi(sec);
   }

void getime(int *hour, int *min, int *sec)
        {
   struct tm *t;
   time_t ltime;
   time(&ltime);
   t=localtime(&ltime);
   *hour=t->tm_hour; /* tunti */
   *min=t->tm_min;   /* minuutti */
   *sec=t->tm_sec;   /* sekunti */
   }

void gedate(int *year, int *month, int *day)
   {
   struct tm *t;
   time_t ltime;
   time(&ltime);
   t=localtime(&ltime);
   *day=t->tm_mday;     /* p{iv{ */
   *month=t->tm_mon+1;  /* kuukausi */
   *year=1900+t->tm_year;    /* vuosi */
   }


void datim(int *iy,int *im,int *id,int *ih,int *in,int *is,int *iff)
   {
   gedate(iy,im,id);
   getime(ih,in,is);
   }

void dstot_(int *it1,int *it2,int *it3,int *it4,int *it5,int *it6,long *aday,long *asec)
   {
   long l1,l2,l3,l4,ad;
   double fday=0.,f=0.;

   if(*asec>=M_DAY)
      {
      *aday+=*asec/M_DAY;
      *asec=*asec%M_DAY;
      }
   l1=*asec/M_HOUR;
   l2=*asec-l1*M_HOUR;
   l2=l2/60L;
   l3=*asec-l1*M_HOUR-l2*M_MIN;
   *it4=(int)(l1);
   *it5=(int)(l2);
   *it6=(int)(l3);

   ad=*aday;
   *it1=1;
   while(ad>0)
      {
      if(leap_year(*it1)) ad-=366L; else ad-=365L;
      ++*it1;
      }
   --*it1;
   if(leap_year(*it1)) ad+=366L; else ad+=365L;

   if(leap_year(*it1))month_lenghts[1]=29;
   *it2=1;
   i1=0;
   while((i1+month_lenghts[*it2-1])<(int)ad)
      {
      i1+=month_lenghts[*it2-1];
      ++*it2;
      }
   *it3=(int)ad-i1;
   month_lenghts[1]=28;
   }

void ttods_(float *at1,float *at2,float *at3,float *at4,float *at5,float *at6,long *aday,long
*asec)
   {
   long l1=0L,l2=0L;
   double fsec=0.,f=0.;
   int yr;
   fsec=(double)(*at6)+M_MIN*(double)(*at5)+M_HOUR*(double)(*at4);
   if(fsec>=M_DAY)
      {
      f=fsec/M_DAY;
      *at3+=(float)((long)f);
      fsec-=((float)(long)f)*M_DAY; /* tarkista */
      *at6=(float)fsec;
      }
   *asec=(long)fsec;
   *aday=(long)*at3;
   for(i1=1;i1<nintx(at2);i1++)
      *aday+=(double)month_lenghts[i1-1];
   if(leap_year(nintx(at1)) && nintx(at2)>=3) *aday += 1L;
   for(yr=1;yr<nintx(at1);yr++)
      {
```

```
      if(leap_year(yr))*aday+=366L;
      else *aday+=365L;
      }
   }

int nintx(float *f)
   {
   if(*f>32767.)return(32767);
   if(*f<-32767.)return(-32767);
   return((int)(*f+0.5));
   }

long longnint(float *f)
   {
   if(*f>2147483647.0f)return(2147483647L);
   if(*f<-2147483647.0f)return(-2147483647L);
   return((long)(*f+0.5));
   }

int leap_year(int year)
   {
   if(year%4!=0)return(0);
   if(year%100==0 && year%400!=0)return(0);
   else return(1);
   }

time_t itdiff(time_t long_time)
   {
   long tdiff;
   tzset();
   tdiff = timezone;
   if (daylight != 0 )
      if(in_dst(long_time))
         tdiff = tdiff - 3600;
   return(tdiff);
   }

int in_dst(time_t long_time)
   {
   if(european_dst==1)
      return(in_dst_e(long_time));
   else
      return(in_dst_a(long_time));
   }

int in_dst_a(time_t long_time) /* American */
   {
   int m1,w1,md,h1;
   tzset();
   m1=month(long_time);
   w1=weekday(long_time);
   md=mday(long_time);
   h1=hours(long_time);
   if(m1<3 || m1>9) return(0);
   if(m1>3 && m1<9) return(1);
   if(m1==3) /* April */
      {
      if(year(long_time)>1986)
           {
           if(md<=7&&w1==0&&h1<2)return(0);
            else return(1);
           if(md<=7&&md-1<w1)return(0);
           return(1);
           }
      else
           {
           if(md>=24&&w1==0&&h1<2)return(0);
            else return(1);
           if(md>=24&&30-md<7-w1)return(0);
           return(1);
           }
      }
   else    /* October */
      {
      if(md>=25&& w1==0&&h1>1)return(0);
      else return(1);
      if(md>=25&& 31-md<7-w1)return(0);
      return(1);
      }
   }

int weekday(time_t long_time){return(gmtime(&long_time)->tm_wday);}
int mday(time_t long_time)   {return(gmtime(&long_time)->tm_mday);}
int month(time_t long_time)  {return(gmtime(&long_time)->tm_mon);}
int year(time_t long_time)   {return(gmtime(&long_time)->tm_year+1900);}
int hours(time_t long_time)  {return(gmtime(&long_time)->tm_hour);}
int yday(time_t long_time)   {return(gmtime(&long_time)->tm_yday);}

int in_dst_e(time_t long_time) /* European */
   {
   int m1,w1,md,h1;
   tzset();
   m1=month(long_time);
   w1=weekday(long_time);
   md=mday(long_time);
   h1=hours(long_time);
   if(m1<2 || m1>8) return(0);
   if(m1>2 && m1<8) return(1);
   if(m1==2) /* March */
      {
          if(md>=25&&w1==0&&h1<4)return(0);
      else return(1);
          if(md>=25&&31-md<7-w1)return(0);
          return(1);
      }
   else    /* October */
      {
      if(md>=24&& w1==0&&h1>3)return(0);
      else return(1);
      if(md>=24&& 30-md<7-w1)return(0);
      return(1);
      }
   }

void tlongt(time_t long_time, int *ye, int *mo, int *da, int *ho, int *mi, int *se)
   {
   *ye = gmtime(&long_time)->tm_year+1900;
   *mo = gmtime(&long_time)->tm_mon+1;
   *da = gmtime(&long_time)->tm_mday;
   *ho = gmtime(&long_time)->tm_hour;
   *mi = gmtime(&long_time)->tm_min;
   *se = gmtime(&long_time)->tm_sec;
   }

void socinf_(int *j1, int *j2, int *j3, int *j4, int *j5, int *j6)
   {
   *j1=specta_1.isocy;
   *j2=specta_1.isocm;
   *j3=specta_1.isocd;
   *j4=specta_1.isoch;
   *j5=specta_1.isocmi;
   *j6=specta_1.isocs;
   }

void eocinf_(int *j1, int *j2, int *j3, int *j4, int *j5, int *j6)
   {
   *j1=specta_1.ieocy;
   *j2=specta_1.ieocm;
   *j3=specta_1.ieocd;
   *j4=specta_1.ieoch;
   *j5=specta_1.ieocmi;
   *j6=specta_1.ieocs;
   }

void alivtm_(float *f)
  {
    *f=specta_1.tlive;
  }

void calcuate_start_time(int year,int date,long seconds)
```

```
   {
   int month,day,hour,min,sec;

   specta_1.isocy=year;
   specta_1.isocm=1;
   specta_1.isocd=1;
   specta_1.isoch=0;
   specta_1.isocmi=0;
   specta_1.isocs=0;

   if(date>366) return;
   if(seconds>86400L) return;

   if(leap_year(year))month_lenghts[1]=29; else month_lenghts[1]=28;
   day=date;
   month=0;
   while(day>month_lenghts[month])
      {
      day-=month_lenghts[month];
      month++;
      }
   specta_1.isocm=month+1;
   specta_1.isocd=day;

   hour=(int)(seconds/3600L);
   min=(int)((seconds%3600L)/60L);
   sec=(int)((seconds%3600L)%60L);
   specta_1.isoch=hour;
   specta_1.isocmi=min;
   specta_1.isocs=sec;

   month_lenghts[1]=28;
   }

void set_now_soc(void)
   {
   int hour,min,sec,year,month,day;
   getime(&hour, &min, &sec);
   gedate(&year, &month, &day);
   specta_1.isocy= year;
   specta_1.isocm= month;
   specta_1.isocd= day;
   specta_1.isoch= hour;
   specta_1.isocmi= min;
   specta_1.isocs= sec;
   }

int month_number(char *month)
   {
   int i;
   char cmon[4];
   strncpy(cmon,month,3);
   cmon[3]='\0';
   for (i=0;i<12;i++)
      {
      if(strcmpi(cmon,months[i])==0) return(i+1);
      }
   return(0);
   }

int number_month(int num, char *month)
   {
   strcpy(month,months[num-1]);
   return(0);
   }

int chk_leap(int iy)
        {
        if( (iy%4==0 && iy%100!=0 ) || iy%400==0) return(1);
        else return(0);
        }

int datchk(int iy,int im,int id)
        {
        int leap;
        leap=chk_leap(iy);
        if(leap && im ==1) leap=1; else leap=0;
        if(iy>0 && im>0 && im<=12 && id>0 && id <=month_lenghts[im-1]+leap) return(1);
        else return(0);
        }

int timchk(int ih, int im, int is)
        {
        if(ih>=0 && ih <24 && im>=0 && im<60 && is>=0 && is<60) return(1);
        else return(0);
        }

void labsoc(int i1, int i2, int i3, int i4, int i5, int i6)
        {
        if(!datchk(i1,i2,i3))
                {
                error_out("INCORRECT SOC DATE");
                return;
                }
        if(!timchk(i4,i5,i6))
                {
                error_out("INCORRECT SOC TIME");
                return;
                }
        specta_1.isocy=i1;
        specta_1.isocm=i2;
        specta_1.isocd=i3;
        specta_1.isoch=i4;
        specta_1.isocmi=i5;
        specta_1.isocs=i6;
        specta_1.isoc=1;
        verify_date(specta_1.isocy,specta_1.isocm,specta_1.isocd);
        }

void labeoc(int i1, int i2, int i3, int i4, int i5, int i6)
        {
        if(!datchk(i1,i2,i3))
                {
                error_out("INCORRECT EOC DATE");
                return;
                }
        if(!timchk(i4,i5,i6))
                {
                error_out("INCORRECT EOC TIME");
                return;
                }
        specta_1.ieocy=i1;
        specta_1.ieocm=i2;
        specta_1.ieocd=i3;
        specta_1.ieoch=i4;
        specta_1.ieocmi=i5;
        specta_1.ieocs=i6;
        specta_1.ieoc=1;
        verify_date(specta_1.ieocy,specta_1.ieocm,specta_1.ieocd);
        }

void labsoi(int i1, int i2, int i3, int i4, int i5, int i6)
        {
        if(!datchk(i1,i2,i3))
                {
                error_out("INCORRECT SOI DATE");
                return;
                }
        if(!timchk(i4,i5,i6))
                {
                error_out("INCORRECT SOI TIME");
                return;
                }
        specta_1.isoiy=i1;
        specta_1.isoim=i2;
        specta_1.isoid=i3;
        specta_1.isoih=i4;
        specta_1.isoimi=i5;
        specta_1.isois=i6;
        specta_1.isoi=1;
        verify_date(specta_1.isoiy,specta_1.isoim,specta_1.isoid);
        }

void labeoi(int i1, int i2, int i3, int i4, int i5, int i6)
        {
        if(!datchk(i1,i2,i3))
                {
                error_out("INCORRECT EOI DATE");
                return;
                }
        if(!timchk(i4,i5,i6))
                {
                error_out("INCORRECT EOI TIME");
                return;
                }
        specta_1.ieoiy=i1;
        specta_1.ieoim=i2;
        specta_1.ieoid=i3;
        specta_1.ieoih=i4;
        specta_1.ieoimi=i5;
        specta_1.ieois=i6;
        specta_1.ieoi=1;
        verify_date(specta_1.ieoiy,specta_1.ieoim,specta_1.ieoid);
        }

void labpre(int i1, int i2, int i3, int i4, int i5, int i6)
        {
        if(!datchk(i1,i2,i3))
                {
                error_out("INCORRECT PRE DATE");
                return;
                }
        if(!timchk(i4,i5,i6))
                {
                error_out("INCORRECT PRE TIME");
                return;
                }
        specta_1.iprey=i1;
        specta_1.iprem=i2;
        specta_1.ipred=i3;
        specta_1.ipreh=i4;
        specta_1.ipremi=i5;
        specta_1.ipres=i6;
        specta_1.ipre=1;
        verify_date(specta_1.iprey,specta_1.iprem,specta_1.ipred);
        }

void update_times(void)
    {
    static int j1,j2,j3,j4,j5,j6;
    static long l1,l2,l3,l4,l5,l6;
    float g1,g2,g3,g4,g5,g6;

g1=specta_1.isoiy;g2=specta_1.isoim;g3=specta_1.isoid;g4=specta_1.isoih;g5=specta_1.isoimi;g6=spe
cta_1.isois;
    ttods_(&g1,&g2,&g3,&g4,&g5,&g6,&l1,&l2);

g1=specta_1.ieoiy;g2=specta_1.ieoim;g3=specta_1.ieoid;g4=specta_1.ieoih;g5=specta_1.ieoimi;g6=spe
cta_1.ieois;
    ttods_(&g1,&g2,&g3,&g4,&g5,&g6,&l3,&l4);

g1=specta_1.isocy;g2=specta_1.isocm;g3=specta_1.isocd;g4=specta_1.isoch;g5=specta_1.isocmi;g6=spe
cta_1.isocs;
    ttods_(&g1,&g2,&g3,&g4,&g5,&g6,&l5,&l6);

    specta_1.twait=86400.0*(float)(l5-l3)+(float)(l6-l4);
    specta_1.texp=86400.0*(float)(l3-l1)+(float)(l4-l2);


g1=specta_1.isocy;g2=specta_1.isocm;g3=specta_1.isocd;g4=specta_1.isoch;g5=specta_1.isocmi;g6=spe
cta_1.isocs;
    ttods_(&g1,&g2,&g3,&g4,&g5,&g6,&l1,&l4);
    l2=(long)(specta_1.treal)/86400L;
    l5=(long)(specta_1.treal)%86400L;
    l3=l1+l2;
    l6=l4+l5;

    dstot_(&j1,&j2,&j3,&j4,&j5,&j6,&l3,&l6);
    specta_1.ieocy=j1;
    specta_1.ieocm=j2;
    specta_1.ieocd=j3;
    specta_1.ieoch=j4;
    specta_1.ieocmi=j5;
    specta_1.ieocs=j6;
    }

void verify_date(int y, int m, int d)
    {
    }

void ortecdate(char *ch, int *y,int *m, int *d)
   {

   char localtim[20];
   char is2000;
   sprintf(localtim,"%s",ch);
   is2000=localtim[9];
   localtim[2]='\0';
   localtim[6]='\0';
   localtim[9]='\0';
   *d=atoi(localtim);
   *m=month_number(&localtim[3]);
   if(*m==0)*m=1;
   *y=1900+atoi(&localtim[7]);
   if(is2000=='1')*y+=100;
   }

void ortectime(char *ch, int *h,int *m, int *s)
   {
   char localtim[20];
   sprintf(localtim,"%s",ch);
   localtim[2]='\0';
   localtim[5]='\0';
   localtim[8]='\0';
   *h=atoi(localtim);
   *m=atoi(&localtim[3]);
   *s=atoi(&localtim[6]);
   }

void dateortec(char *ch, int *y,int *m, int *d)
   {
   itodatortec(y,m,d,ch,10);
   }

void timeortec(char *ch, int *h,int *m, int *s)
   {
   itotim(h,m,s,ch,8);
   }
```

RMS.C

```
/*
CONVSPE RMS and ASC file format
*/

#define strcasecmp stricmp
#include "all.h"
char buffer[256],bb[10],bbb[20],tmpname1[256],tmpname2[256];

#define UNKNOWN 0
#define SPECTRUM 1
#define ENERGY 2
#define EFFICIENCY 3
#define RESOLUTION 4
#define SHAPE 5
#define TOTALEFF 6
#define ANNIHILATION 7
#define QUANTITY 15
```

```
#define ACQUISITION 16
#define ENERGYCOEF 20
#define RESOLUTIONCOEF 21
#define EFFICIENCYCOEF 22
#define QA 18
#define CERTIFICATE 19
#define PEAKTABLE 20
#define BASELINE 21

int rms_read_spectrum(char *fname, int sum)
   {
   FILE *readable;
   int iret,i,n;
   int j1,j2,j3,j4,j5,j6;
   int line, type;
   int maxchan;
   float channel,energy;
   float channel1,energy1;
   float channel2,energy2;
   float fwhmen, fwhmch;
   float s1,s2,s3,s4,s5;
   int i1, i2;


   readable = fopen(fname,"r");
   type=UNKNOWN;
   if(readable)
      {
      while(!feof(readable))
         {
         get_file_line(readable,buffer,255);
         if(buffer[0]=='#')
            {
            strncpy(bb,buffer,6);
            bb[5]='\0';
           if(strlen(buffer)>10)
                {
               strncpy(bbb,buffer,10);
               bbb[10]='\0';
                }

            type=UNKNOWN;

            if(strcasecmp(bb,"#Spec")==0)
               {
               type=SPECTRUM;
               get_file_line(readable,buffer,255);
               buffer[5]='\0';
               maxchan=atoi(buffer);
               specta_1.nchanl=maxchan;
               line=0;
               }

            if(strcasecmp(bb,"#Head")==0)
               {
               get_file_line(readable,buffer,255);
               }

            if(strcasecmp(bb,"#Acqu")==0)
               {
               get_file_line(readable,buffer,255);
               sscanf(buffer,"%s %s %f %f", tmpname1,tmpname2,&s1,&s2);

              buffer[21]='\0';
              dttoi(&j1,&j2,&j3,&j4,&j5,&j6,buffer,22);
               labsoc(j1,j2,j3,j4,j5,j6);
              if(!sum)
                {
                 specta_1.treal=s1;
                 specta_1.tlive=s2;
                 }
               else
                 {
                specta_1.treal+=s1;
                 specta_1.tlive+=s2;
                 }

               }

            if(strcasecmp(bb,"#Samp")==0)
               {
               get_file_line(readable,buffer,255);
               buffer[5]='\0';
               /* specta_1.alabel[0]=atof(&buffer[0]); */
               }

            if(strcasecmp(bb,"#Coll")==0)
               {
               get_file_line(readable,buffer,255);
               specta_1.iset=1;

               buffer[21]='\0';
               dttoi(&j1,&j2,&j3,&j4,&j5,&j6,&buffer[0],20);
               labsoi(j1,j2,j3,j4,j5,j6);

               buffer[42]='\0';
               dttoi(&j1,&j2,&j3,&j4,&j5,&j6,&buffer[22],20);
               labeoi(j1,j2,j3,j4,j5,j6);

               specta_1.svol=atof(&buffer[44]);
              specta_1.smass=1.0;
               }
            }
         else
            {

            if(type==SPECTRUM)
               {
              if(sum)
                 {
                 sscanf(buffer,"%d %f %f %f %f %f", &i1,&s1,&s2,&s3,&s4,&s5);
                 specta_1.spec[line]  +=s1;
                 specta_1.spec[line+1]+=s2;
                 specta_1.spec[line+2]+=s3;
                 specta_1.spec[line+3]+=s4;
                 specta_1.spec[line+4]+=s5;
                 line+=5;
                 }
              else
                 {
                 sscanf(buffer,"%d %f %f %f %f %f", &i1,
                 &specta_1.spec[line],
                 &specta_1.spec[line+1],
                 &specta_1.spec[line+2],
                 &specta_1.spec[line+3],
                 &specta_1.spec[line+4]);
                 line+=5;
                 /*specta_1.nchanl=line;*/
                 }
               }
            }
         }

      j1=specta_1.ieoiy;
      j2=specta_1.ieoim;
      j3=specta_1.ieoid;
      j4=specta_1.ieoih;
      j5=specta_1.ieoimi;
      j6=specta_1.ieois;
      itodt(&j1,&j2,&j3,&j4,&j5,&j6,tmpname1,22);

      sprintf(specta_1.title,"CONVSPE converted file.");

      specta_1.spec[specta_1.nchanl]=0;
      fclose(readable);
      return(1);
      }
   return(0);
   }
```

```
int rms_read_calibr(char *fname, int sum)
   {
   FILE *readable;
   int iret,i,n;
   int j1,j2,j3,j4,j5,j6;
   int line, type;
   int maxchan;
   float channel,energy;
   float channel1,energy1;
   float channel2,energy2;
   float fwhmen, fwhmch;
   float s1,s2,s3,s4,s5;
   int i1, i2;

   char buffer[256],bb[10],bbb[20];
   readable = fopen(fname,"r");
   type=UNKNOWN;

   if(readable)
      {
      cshape_1.anhcp=0.0;
      cshape_1.anhcl=0.0;
      cshape_1.anhch=0.0;
      cshape_1.anhcw=0.0;
      cshape_1.anhcpe=0.0;
      cshape_1.anhcle=0.0;
      cshape_1.anhche=0.0;
      cshape_1.anhcwe=0.0;

      while(!feof(readable))
         {
         get_file_line(readable,buffer,255);

         if(buffer[0]=='#')
            {
            strncpy(bb,buffer,6);
            bb[5]='\0';
           if(strlen(buffer)>10)
                {
               strncpy(bbb,buffer,10);
               bbb[10]='\0';
                }

            type=UNKNOWN;

            if(strcasecmp(bb,"#Ener")==0)
               {
               if(strcasecmp(bbb,"#Energy_co")==0)
                  {
                  type=ENERGYCOEF;
                   }
              else
                  {
                  type=ENERGY;
                  cenerg_1.neread=0;
                  cenerg_1.ifunce=1;
                  line=0;
                  }
               }

            if(strcasecmp(bb,"#Tota")==0)
               {
               type=TOTALEFF;
               ctotf_1.ntread=0;
               ctotf_1.ifunct=1;
               ctotf_1.idegrt=1;
               line=0;
               }

            if(strcasecmp(bb,"#Shap")==0)
               {
              type=SHAPE;
               cshape_1.nsread=0;
               cshape_1.ifunsw=1;
               cshape_1.ifunsh=1;
               cshape_1.ifunsl=1;
               line=0;
               }
            if(strcasecmp(bb,"#Anni")==0)
               {
              type=ANNIHILATION;
               line=0;
               }

            if(strcasecmp(bb,"#Effi")==0)
               {
               if(strcasecmp(bbb,"#Effi_coef")==0)
                  {
                  type=EFFICIENCYCOEF;
                  }
              else
                  {
                  type=EFFICIENCY;
                  cefff_1.nfread=0;
                  cefff_1.ifuncd=1;
                  cefff_1.idegrd=1;
                  line=0;
                  }
               }

            if(strcasecmp(bb,"#Reso")==0)
               {
               if(strcasecmp(bbb,"#Reso_coef")==0)
                  {
                  type=RESOLUTIONCOEF;
                  }
              else
                  {
                 type=RESOLUTION;
                  cshape_1.nsread=0;
                  cshape_1.ifunsw=1;
                  cshape_1.ifunsh=1;
                  cshape_1.ifunsl=1;
                  line=0;
                 }
               }

            }
         else
            {

            if(type==EFFICIENCY)
               {
               sscanf(buffer,"%f %f %f",
               &cefff_1.eneffi[cefff_1.nfread],
               &cefff_1.effici[cefff_1.nfread],
               &cefff_1.efferr[cefff_1.nfread]);
               cefff_1.nfread++;
               }

            if(type==TOTALEFF)
              {
               sscanf(buffer,"%f %f %f",
               &ctotf_1.enetfi[ctotf_1.ntread],
               &ctotf_1.eftici[ctotf_1.ntread],
               &ctotf_1.efterr[ctotf_1.ntread]);
               ctotf_1.ntread++;
               }

            if(type==ENERGY)
               {
               sscanf(buffer,"%f %f %f",
               &cenerg_1.energy[cenerg_1.neread],
               &cenerg_1.channl[cenerg_1.neread],
               &cenerg_1.enerro[cenerg_1.neread]);

              cenerg_1.channl[cenerg_1.neread]+=1.0;

               cenerg_1.neread++;
               }
```

```
            if(type==SHAPE)
               {
               sscanf(buffer,"%f %f %f %f %f %f %f",
                     &cshape_1.cpread[cshape_1.nsread],
                     &cshape_1.cwread[cshape_1.nsread],
                     &cshape_1.cwerro[cshape_1.nsread],
                     &cshape_1.clread[cshape_1.nsread],
                     &cshape_1.clerro[cshape_1.nsread],
                     &cshape_1.chread[cshape_1.nsread],
                     &cshape_1.cherro[cshape_1.nsread]);
               cshape_1.nsread++;
               }

            if(type==ANNIHILATION)
               {
               sscanf(buffer,"%f %f %f %f %f %f %f %f",
                     &cshape_1.anhtol,
                     &cshape_1.anhcp,
                     &cshape_1.anhcw,
                     &cshape_1.anhch,
                     &cshape_1.anhcl,
                     &cshape_1.anhcwe,
                     &cshape_1.anhche,
                     &cshape_1.anhcle);

               }

            if(type==RESOLUTION)
               {
               sscanf(buffer,"%f %f %f" , &energy, &fwhmen, &cshape_1.cwerro[cshape_1.nsread]);

               cshape_1.cpread[cshape_1.nsread]=energy;
               cshape_1.cwread[cshape_1.nsread]=fwhmen;

               cshape_1.clread[cshape_1.nsread]=cshape_1.cwread[cshape_1.nsread]*2;
               cshape_1.chread[cshape_1.nsread]=cshape_1.cwread[cshape_1.nsread]*2;
               cshape_1.cherro[cshape_1.nsread]=cshape_1.cwerro[cshape_1.nsread];
               cshape_1.clerro[cshape_1.nsread]=cshape_1.cwerro[cshape_1.nsread];
               cshape_1.nsread++;
               }

            }
         }

      return(1);
      }
   return(0);
   }


void get_file_line(FILE *fptr, char *line, int len)
   {
   char ch=0;
   int len_here=0;
   *line='\0';
   while(ch!='\n' && !feof(fptr) )
      {
      ch=fgetc(fptr);
      if(len_here<len && ch!='\n')
         {
         *line=ch;
         line++;
         *line='\0';
         }
      len_here++;
      }
   }


int rms_write_spectrum(char *fname, int sum)
   {
   FILE *writeable;
   int i;
   int j1,j2,j3;
   char c1[128],c2[128],c3[128],c4[128];
   writeable = fopen(fname,"w");

   if(writeable)
      {
      fprintf(writeable,"BEGIN RMS2.0\n");
      fprintf(writeable,"MSG_TYPE DATA\n");
      fprintf(writeable,"MSG_ID 99999\n");
      fprintf(writeable,"DATA_TYPE SAMPLEPHD\n");
      fprintf(writeable,"#Header\n%s\n",specta_1.title);
      fprintf(writeable,"#Comment\n");
      fprintf(writeable,"%s\n",run.version);
      fprintf(writeable,"#Collection\n");
           j1=specta_1.isoiy;
           j2=specta_1.isoim;
           j3=specta_1.isoid;
           itodat(&j1,&j2,&j3,c1,11);
           j1=specta_1.isoih;
           j2=specta_1.isoimi;
           j3=specta_1.isois;
           itotim(&j1,&j2,&j3,c2,8);
           j1=specta_1.ieoiy;
           j2=specta_1.ieoim;
           j3=specta_1.ieoid;
           itodat(&j1,&j2,&j3,c3,11);
           j1=specta_1.ieoih;
           j2=specta_1.ieoimi;
           j3=specta_1.ieois;
           itotim(&j1,&j2,&j3,c4,8);
      fprintf(writeable,"%s %s  %s %s %.2f\n",c1,c2,c3,c4, specta_1.svol);
      fprintf(writeable,"#Acquisition\n");
           j1=specta_1.isocy;
           j2=specta_1.isocm;
           j3=specta_1.isocd;
           itodat(&j1,&j2,&j3,c1,12);
           j1=specta_1.isoch;
           j2=specta_1.isocmi;
           j3=specta_1.isocs;
           itotim(&j1,&j2,&j3,c2,12);
      fprintf(writeable,"%s %s %.2f       %.2f\n",c1,c2, specta_1.treal, specta_1.tlive);
      fprintf(writeable,"#Energy\n");
      for(i=0;i<cenerg_1.neread;i++)
         fprintf(writeable,"%16.5f %16.5f %16.5f\n",cenerg_1.energy[i], cenerg_1.channl[i],
cenerg_1.enerro[i]);

      fprintf(writeable,"#Resolution\n");
      for(i=0;i<cshape_1.nsread;i++)
         fprintf(writeable,"%16.5f %16.5f %16.5f\n",cshape_1.cpread[i], cshape_1.cwread[i],
cshape_1.cwerro[i]);

      fprintf(writeable,"#Efficiency\n");
      for(i=0;i<cefff_1.nfread;i++)
         fprintf(writeable,"%16.7f %16.7f %16.7f\n",cefff_1.eneffi[i], cefff_1.effici[i],
cefff_1.efferr[i]);

      fprintf(writeable,"#Spectrum\n");
      fprintf(writeable,"%d 0\n",specta_1.nchanl);
      for(i=0;i<specta_1.nchanl;i+=5)
          {
         fprintf(writeable,"%-5d %10.0f %10.0f %10.0f %10.0f
%10.0f\n",i,specta_1.spec[i],specta_1.spec[i+1],specta_1.spec[i+2],specta_1.spec[i+3],specta_1.sp
ec[i+4]);
          }
      fprintf(writeable,"STOP");
      fclose(writeable);
      }
   else
      return(0);
   return(1);
   }


int asc_read_spectrum(char *fname, int sum)
   {
   FILE *readable;
   int i=0;

   readable = fopen(fname,"r");
   if(readable)
      {
      while(!feof(readable))
         {
         get_file_line(readable,buffer,255);
         specta_1.spec[i]=atof(buffer);
         i++;
         }
      fclose(readable);
      specta_1.nchanl=i+1;
      specta_1.iset=1;
      return(1);
      }
   return(0);
   }



int asc_write_spectrum(char *fname, int sum)
   {
   FILE *writeable;
   int i;
   writeable = fopen(fname,"w");

   if(writeable)
      {
      for(i=0;i<specta_1.nchanl;i++)
          {
              fprintf(writeable,"%.0f\n",specta_1.spec[i]);
              }
      fclose(writeable);
      }
   else
      return(0);
   return(1);
   }
```

SPC.C

```
/*
CONVSPE SPC file format
*/

#include "all.h"
char readrec[65536];

int spc_read_spectrum(char *fname, int sum)
   {
   FILE *readable;
   int i,j;
   int inftyp;
   int filtyp;
   int acqirp;
   int effprp;
   int eneprp;
   int detdrp;
   int sperec;
   int speprp;
   int numchan;
   int enenum;
   int effnum;
   int idum;
   int c1;
   long *ip;
   char tim1[20];
   float *acqtim,*livetime,*realtime,*fp,*fp2,*fp3;
   double *date2;

   sprintf(specta_1.title,"SPC file read in from file: %s",fname);
   readable = fopen(fname,"rb");
```

```
if(readable)
   {
   j=0;
   while(!feof(readable))
           {
      for(i=0;i<128;i++)
              {
              readrec[i+j*128]=fgetc(readable);
              }
      j++;
           }
   fclose(readable);
   }
else
   return(0);

for(i=0;i<2800;i++)
   {
   printf(" i: %4d  %6d   ",i+1,get_i2(&readrec[i*2]));
   if(i%4==0) printf("\n");
   }
inftyp=get_i2(&readrec[0]);
filtyp=get_i2(&readrec[2]);
acqirp=get_i2(&readrec[8])-1;
effprp=get_i2(&readrec[38])-1;
eneprp=get_i2(&readrec[42])-1;
enenum=get_i2(&readrec[44])-1;
speprp=get_i2(&readrec[60])-1;
effnum=get_i2(&readrec[58])-1;
sperec=get_i2(&readrec[62])-1;
numchan=get_i2(&readrec[64]);
realtime=(float *)&readrec[90];
livetime=(float *)&readrec[94];
acqtim=(float *)&readrec[68];
date2=(double *)&readrec[72];
specta_1.iset=1;
specta_1.nchanl=numchan;
specta_1.treal=*realtime;
specta_1.tlive=*livetime;

strncpy(tim1,&readrec[acqirp*128+16],10);
tim1[10]='\0';
ortecdate(tim1,&specta_1.isocy,&specta_1.isocm,&specta_1.isocd);

strncpy(tim1,&readrec[acqirp*128+28],8);
tim1[8]='\0';
ortectime(tim1,&specta_1.isoch,&specta_1.isocmi,&specta_1.isocs);

strncpy(tim1,&readrec[acqirp*128+92],10);
tim1[10]='\0';
ortecdate(tim1,&specta_1.isoiy,&specta_1.isoim,&specta_1.isoid);

strncpy(tim1,&readrec[acqirp*128+102],8);
tim1[8]='\0';
ortectime(tim1,&specta_1.isoih,&specta_1.isoimi,&specta_1.isois);

strncpy(tim1,&readrec[acqirp*128+110],10);
tim1[10]='\0';
ortecdate(tim1,&specta_1.ieoiy,&specta_1.ieoim,&specta_1.ieoid);

strncpy(tim1,&readrec[acqirp*128+120],8);
tim1[8]='\0';
ortectime(tim1,&specta_1.ieoih,&specta_1.ieoimi,&specta_1.ieois);
update_times();
if(filtyp!=1 && filtyp!=5)
   {
   error_out("SPC: only filetypes 1 and 5 are supported\n");
   exit(0);
   }
for(i=0;i<16;i++)
   {
   fp=(float *)&readrec[eneprp*128+i*4];
   fp2=(float *)&readrec[(eneprp+1)*128+i*4];
   fp3=(float *)&readrec[(eneprp+2)*128+i*4];
   cenerg_1.energy[i]=*fp2;
```

```
      cenerg_1.enerro[i]=0.1;
      cenerg_1.channl[i]=*fp;
      cshape_1.cpread[i]=*fp2;
      cshape_1.cwread[i]=*fp3;
      cshape_1.cwerro[i]=*fp3 * 0.05;

      if(*fp>0.0)cshape_1.cwread[i]*=(*fp2 / *fp);

      if(*fp>0.0)
         {
         cenerg_1.neread=i+1;
         cshape_1.nsread=i+1;
         }
      }
   cenerg_1.neread=enenum;
   cshape_1.nsread=enenum;

   for(i=0;i<16;i++)
      {
      fp=(float *)&readrec[effprp*128+i*8];
      fp2=(float *)&readrec[effprp*128+i*8+4];
      cefff_1.effici[i]=*fp2;
      cefff_1.efferr[i]=*fp2 * 0.05;
      cefff_1.eneffi[i]=*fp;
      if(*fp>0.0)cefff_1.nfread=i+1;
      }
   cefff_1.nfread=effnum;
   for(i=0;i<numchan;i++)
           {
           if(filtyp==5)
             {
             fp=(float *)&readrec[speprp*128+i*4];
             specta_1.spec[i]=*fp;
             }
           if(filtyp==1)
             {
             ip=(long *)&readrec[speprp*128+i*4];
             specta_1.spec[i]=(float)*ip;
             }
           }
   return(1);
   }

put_i2(char *r,int i)
   {
   int *ip;
   ip=(int *)r;
   *ip=i;
   }

int get_i2(char *r)
   {
   char c1,c2;
   c1=*r;
   r++;
   c2=*r;
   return(256*(int)c2+(int)c1);
   }


int spc_write_spectrum(char *fname, int sum)
   {
   FILE *writeable;
   long lastrec;
   int i,j;
   long l;
   int inftyp;
   int filtyp;
   int acqirp;
   int effprp;
   int eneprp;
   int detdrp;
   int sperec;
   int speprp;
   int numchan;
   int enenum;
   int effnum;
   int idum;
   int c1;
   long *ip;
   char tim1[20];
   float *acqtim,*livetime,*realtime,*fp,*fp2,*fp3;

      put_i2(&readrec[0],1);
      put_i2(&readrec[2],5);
      put_i2(&readrec[4],1);
      put_i2(&readrec[6],0);

      put_i2(&readrec[8],2); /* acq. par */
      put_i2(&readrec[10],3);
      put_i2(&readrec[12],4);
      put_i2(&readrec[14],0);

      put_i2(&readrec[16],14);
      put_i2(&readrec[18],20);
      put_i2(&readrec[20],29);
      put_i2(&readrec[22],39);

      put_i2(&readrec[16],0);
      put_i2(&readrec[18],0);
      put_i2(&readrec[20],0);
      put_i2(&readrec[22],0);

      put_i2(&readrec[24],0);
      put_i2(&readrec[26],0);
      put_i2(&readrec[28],0);
      put_i2(&readrec[30],0);

      put_i2(&readrec[32],9);
      put_i2(&readrec[34],5);
      put_i2(&readrec[36],6);
      put_i2(&readrec[38],13); /* eff cal */

      put_i2(&readrec[40],41); /* ene and reso cal num*/
      put_i2(&readrec[42],10); /* ene and reso cal */
      put_i2(&readrec[44],3); /* ene and reso cal num*/
      put_i2(&readrec[46],0); /* ene and reso cal num*/

      put_i2(&readrec[48],0);
      put_i2(&readrec[50],0);
      put_i2(&readrec[52],38);
      put_i2(&readrec[54],85);


      put_i2(&readrec[56],85);
      put_i2(&readrec[58],1); /* spe */
      put_i2(&readrec[60],44); /* spe */
      put_i2(&readrec[62],specta_1.nchanl/32); /* spe */


      put_i2(&readrec[64],specta_1.nchanl);
      put_i2(&readrec[66],0);
      put_i2(&readrec[68],23202);
      put_i2(&readrec[70],17662);

      put_i2(&readrec[72],7185);
      put_i2(&readrec[74],12754);
      put_i2(&readrec[76],-13452);
      put_i2(&readrec[78],16319);

      put_i2(&readrec[80],0);
      put_i2(&readrec[82],1);
      put_i2(&readrec[84],1);
      put_i2(&readrec[86],0);

      put_i2(&readrec[88],0);
      put_i2(&readrec[90],-12350);
      put_i2(&readrec[92],17852);
      put_i2(&readrec[94],10608);
```

```
put_i2(&readrec[96],17849);



fp=(float *)&readrec[90];
*fp=specta_1.treal;
fp=(float *)&readrec[94];
*fp=specta_1.tlive;
acqirp=2-1;
effprp=13-1;
eneprp=10-1;
speprp=44-1;

dateortec(tim1,&specta_1.isocy,&specta_1.isocm,&specta_1.isocd);
tim1[10]='\0';
strncpy(&readrec[acqirp*128+16],tim1,10);

timeortec(tim1,&specta_1.isoch,&specta_1.isocmi,&specta_1.isocs);
tim1[8]='\0';
strncpy(&readrec[acqirp*128+28],tim1,8);

dateortec(tim1,&specta_1.isoiy,&specta_1.isoim,&specta_1.isoid);
tim1[10]='\0';
strncpy(&readrec[acqirp*128+92],tim1,10);

timeortec(tim1,&specta_1.isoih,&specta_1.isoimi,&specta_1.isois);
tim1[8]='\0';
strncpy(&readrec[acqirp*128+102],tim1,8);

dateortec(tim1,&specta_1.ieoiy,&specta_1.ieoim,&specta_1.ieoid);
tim1[10]='\0';
strncpy(&readrec[acqirp*128+110],tim1,10);

ortectime(tim1,&specta_1.ieoih,&specta_1.ieoimi,&specta_1.ieois);
tim1[8]='\0';
strncpy(&readrec[acqirp*128+120],tim1,8);

if(cenerg_1.neread>16)
   {
   error_out("Warning: over 16 energy calibration points");
   cenerg_1.neread=16;
   }
for(i=0;i<cenerg_1.neread;i++)
   {
   fp=(float *)&readrec[eneprp*128+i*4];
   *fp=cenerg_1.channl[i];
   fp=(float *)&readrec[(eneprp+1)*128+i*4];
   *fp=cenerg_1.energy[i];
   fp=(float *)&readrec[(eneprp+2)*128+i*4];
   *fp=cshape_1.cwread[i]*(cenerg_1.channl[i]/cenerg_1.energy[i]);
   if(cshape_1.cpread[i]!=cenerg_1.energy[i])
       {
       error_out("Warning: energy and FWHM calibration do not match");
       }
   }


if(cefff_1.nfread>16)
   {
   error_out("Warning: over 16 efficiency calibration points");
   cefff_1.nfread=16;
   }
for(i=0;i<cefff_1.nfread;i++)
   {
   fp=(float *)&readrec[effprp*128+i*8];
   *fp=cefff_1.eneffi[i];
   fp=(float *)&readrec[effprp*128+i*8+4];
   *fp=cefff_1.effici[i];
   }
for(i=0;i<specta_1.nchanl;i++)
        {
        fp=(float *)&readrec[speprp*128+i*4];
   *fp=specta_1.spec[i];
        }
   lastrec=speprp*128+specta_1.nchanl*4;
   writeable = fopen(fname,"wb");

   if(writeable)
      {
      for(l=0;l<lastrec;l++)
              fprintf(writeable,"%c",readrec[l]);
      fclose(writeable);
      return(1);
      }
   else
      return(0);
   }
```

CHN.C

```
/*
CONVSPE CHN file format
*/

#include "all.h"

#define MASK 0x7FFFFFFF

int chn_read_spectrum(char *fname, int sum)
   {
   FILE *readable;

   long chan1,chan2;
   int yr;
   int iret,i,j1,j2,j3,j4,j5,j6;
   long treal,tlive;
   char sec[10];
   char min[10];
   char hour[10];
   char day[10];
   char month[10];
   char year[10];
   char cs1;
   char cs2;
   char cs3;
   char cs4;

   readable = fopen(fname,"rb");
   if(readable)
      {
      specta_1.iset=1;
      for(i=0;i<6;i++)
        {
         fread(&cs1,1,1,readable);
              /* printf("%5d  %5d  %c\n",i,cs1,cs1);  */
              }
      fread(&sec,1,2,readable);
      j6=atoi(sec);

      fread(&treal,4,1,readable);
      specta_1.treal=(float)treal/50.0;

      fread(&tlive,4,1,readable);
      specta_1.tlive=(float)tlive/50.0;

      fread(&day,1,2,readable);
      j3=atoi(day);

      fread(&month,1,3,readable);
      j2=month_number(month);

      fread(&year,1,2,readable);
      j1=atoi(year)+1900;

      fread(&year,1,1,readable);
      year[1]='\0';
      if(atoi(year)==1)j1+=100;
      fread(&hour,1,2,readable);
      j4=atoi(hour);

      fread(&min,1,2,readable);
      j5=atoi(min);
/*
      printf("Start of acq.: %d/%d/%d %d:%d:%d\n",j1,j2,j3,j4,j5,j6);
*/
      labsoc(j1,j2,j3,j4,j5,j6);
      labsoi(j1,j2,j3,j4,j5,j6);
      labeoi(j1,j2,j3,j4,j5,j6);

      fread(&cs1,1,1,readable);
      fread(&cs2,1,1,readable);

      fread(&cs1,1,1,readable);
      fread(&cs2,1,1,readable);
      specta_1.nchanl=cs1+256*cs2;
      specta_1.svol=1.0;

      for(i=0;i<specta_1.nchanl;i++)
         {
         fread(&chan1,4,1,readable);
        chan2= MASK & chan1;
        specta_1.spec[i]=(float)chan2;

        }
      fclose(readable);
      update_times();
      return(1);
      }
   else
      return(0);
   }

int chn_write_spectrum(char *fname, int sum)
   {
   FILE *writeable;

   long chan1;
   long  yr;
   int iret,i,j1,j2,j3,j4,j5,j6;
   long treal,tlive;
   char sec[10];
   char min[10];
   char hour[10];
   char day[10];
   char month[10];
   char year[10];
   char cs1;
   char cs2;
   char cs3;
   char cs4;
   writeable = fopen(fname,"wb");
   if(writeable)
      {
      i=-1;
      fwrite(&i,2,1,writeable);
      i=1;
      fwrite(&i,2,1,writeable);
      i=1;
      fwrite(&i,2,1,writeable);

      sec[0]=48+specta_1.isocs/10;
      sec[1]=48+specta_1.isocs%10;
      sec[2]='\0';

      fwrite(&sec[0],1,1,writeable);
      fwrite(&sec[1],1,1,writeable);

      treal=(long)(specta_1.treal*50);
      fwrite(&treal,4,1,writeable);

      tlive=(long)(specta_1.tlive*50);
      fwrite(&tlive,4,1,writeable);
      day[0]=48+specta_1.isocd/10;
      day[1]=48+specta_1.isocd%10;
      day[2]='\0';
      fwrite(&day,1,2,writeable);

      number_month(specta_1.isocm,month);
      fwrite(&month,1,3,writeable);

      yr=specta_1.isocy-1900;
      if(yr<100)
              year[2]='0';
      else
        {
              yr-=100;
              year[2]='1';
         }
      year[0]=48+(yr)/10;
      year[1]=48+(yr)%10;
      year[3]='\0';
      fwrite(&year,1,3,writeable);

      hour[0]=48+specta_1.isoch/10;
      hour[1]=48+specta_1.isoch%10;
      hour[2]='\0';
      fwrite(&hour,1,2,writeable);

      min[0]=48+specta_1.isocmi/10;
      min[1]=48+specta_1.isocmi%10;
      min[2]='\0';
      fwrite(&min,1,2,writeable);
      i=0;
      fwrite(&i,2,1,writeable);

      cs1=specta_1.nchanl%256;
      cs2=specta_1.nchanl/256;

      fwrite(&cs1,1,1,writeable);
      fwrite(&cs2,1,1,writeable);

      for(i=0;i<specta_1.nchanl;i++)
         {
              chan1=(long)specta_1.spec[i];
         fwrite(&chan1,4,1,writeable);
              }
      fclose(writeable);
      return(1);
      }
   else
      return(0);
   }
```

ALL.H

```
/*
CONVSPE all definitions.
*/

#include <stdio.h>
#include <signal.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <malloc.h>
#include <math.h>
#include <float.h>
#include <ctype.h>
#include <stdarg.h>

struct {
    int nchanl;
    float spec[8200];
    int iset;
    char title[256];
    int istype;
    float svol, smass, sheigh, sdim, sdim2, sdim3, sdim4, sdim5, sdim6, tlive,
            treal, tdead, twait, texp, tdec, tpre;
    int     isoc, isocy, isocm, isocd, isoch, isocmi, isocs, ieoc, ieocy,
            ieocm, ieocd, ieoch, ieocmi, ieocs, isoi, isoiy, isoim, isoid,
            isoih, isoimi, isois, ieoi, ieoiy, ieoim, ieoid, ieoih, ieoimi,
            ieois, ipre, iprey, iprem, ipred, ipreh, ipremi, ipres;
} specta_;

#define specta_1 specta_

struct {
    int nfread;
    float eneffi[500], effici[500], efferr[500];
    int iefcon[10], iefmax, iefdt1[7], iefdt2[7];
    float effmin, effmax;
    int ifuncd, idegrd;
    float ceffit[10], edline[2], chised, covmed[4]   /* was [2][2] */,
            chisef, covmef[100]       /* was [10][10] */;
    int kefplo, iefneg, iefchi;
} cefff_;

#define cefff_1 cefff_

struct {
    int neread;
    float channl[500], energy[500], enerro[500];
    int iencon[10], ienmax, iendt1[7], iendt2[7];
    float enemin, enemax;
    int ifunce, idegre;
    float cenfit[10], enline[2], chisel, covmel[4]   /* was [2][2] */,
            chisen, covmen[100]       /* was [10][10] */;
    int kenplo, ienneg, ienchi;
} cenerg_;

#define cenerg_1 cenerg_

struct {
    int nsread;
    float cpread[500], clread[500], chread[500], cwread[500], cperro[500],
            cwerro[500], cherro[500], clerro[500];
    int ishcon[10], ishmax, ishdt1[7], ishdt2[7];
    float anhtol, anhcp, anhcl, anhch, anhcw, shpmin, shpmax, anhcpe, anhcle,
            anhche, anhcwe;
    int ifunsw, idegsw;
    float cswfit[10], chissw, covmsw[100]    /* was [10][10] */;
    int ifunsh, idegsh;
    float cshfit[10], chissh, covmsh[100]    /* was [10][10] */;
    int ifunsl, idegsl;
    float cslfit[10], chissl, covmsl[100]    /* was [10][10] */;
    int kshplo, icwneg, ichneg, iclneg, icwchi, ichchi, iclchi;
    float errdcw, errdcl, errdch;
    int indanh;
    float anhier;
} cshape_;

#define cshape_1 cshape_

struct {
    int ntread;
    float enetfi[500], eftici[500], efterr[500];
    int ietcon[10], ietmax, ietdt1[7], ietdt2[7];
    float eftmin, eftmax;
    int ifunct, idegrt;
    float cetfit[10], etline[2], chiset, covmet[4]   /* was [2][2] */,
            chisft, covmft[100]       /* was [10][10] */;
    int ketplo, ietneg, ietchi;
} ctotf_;

#define ctotf_1 ctotf_

typedef  struct{
       char version[256];
       char copyright[256];
       char disclaimer[256];
```

```
        char distribution[256];
        char license[256];
        char filename[256];
        char peaktable[256];
        char ptffile[256];
        char shoutfil[256];
        char shrglfil[256];
         char runfilename[256];
        float threshold;
        float qathresh;
        int fitmode;
        int baseline;
        float detlevel;
        int beginch;
        int endch;
        float maxsigma;
        float portion;
        int wait;
        }runrecord;

extern runrecord run;

#define ESC 27
int rms_read_spectrum(char *fname, int sum);
int rms_read_calibr(char *fname, int sum);
int rms_write_spectrum(char *fname, int sum);
int asc_read_spectrum(char *fname, int sum);
int asc_write_spectrum(char *fname, int sum);
void get_file_line(FILE *fptr, char *line, int len);
int get_i2(char *r);

int spc_read_spectrum(char *fname, int sum);
int spc_write_spectrum(char *fname, int sum);

int chn_write_spectrum(char *fname, int sum);
int chn_read_spectrum(char *fname, int sum);


/* main.c */
void error_out(char *c);
int main(int argc,char *argv[]);

/* times.c */

void dttoi(int *j1,int *j2,int *j3,int *j4,int *j5,int *j6,char *str,int len);
void itodt(int *j1,int *j2,int *j3,int *j4,int *j5,int *j6,char *str,int len);
void itodt_short(int *j1,int *j2,int *j3,int *j4,int *j5,int *j6,char *str,int len);
void dattoi(int *j1,int *j2,int *j3,char *str,int len);
void itodat(int *j1,int *j2,int *j3,char *str,int len);
void itodatortec(int *j1,int *j2,int *j3,char *str,int len);
void itotim(int *j1,int *j2,int *j3,char *str,int len);
void timtoi(int *j1,int *j2,int *j3,char *str,int len);
void getime(int *hour, int *min, int *sec);
void gedate(int *year, int *month, int *day);
void datim(int *iy,int *im,int *id,int *ih,int *in,int *is,int *iff);
void dstot_(int *it1,int *it2,int *it3,int *it4,int *it5,int *it6,long *aday,long *asec);
void ttods_(float *at1,float *at2,float *at3,float *at4,float *at5,float *at6,long *aday,long
*asec);
int nintx(float *f);
long longnint(float *f);
int leap_year(int year);
time_t itdiff(time_t long_time);
int in_dst(time_t long_time);
int in_dst_a(time_t long_time);
int weekday(time_t long_time);
int mday(time_t long_time);
int month(time_t long_time);
int year(time_t long_time);
int hours(time_t long_time);
int yday(time_t long_time);
int in_dst_e(time_t long_time);
void tlongt(time_t long_time, int *ye, int *mo, int *da, int *ho, int *mi, int *se);
void socinf_(int *j1, int *j2, int *j3, int *j4, int *j5, int *j6);
void eocinf_(int *j1, int *j2, int *j3, int *j4, int *j5, int *j6);
void alivtm_(float *f);
void calcuate_start_time(int year,int date,long seconds);
void set_now_soc(void);
int month_number(char *month);
int number_month(int num, char *month);
int chk_leap(int iy);
int datchk(int iy,int im,int id);
int timchk(int ih, int im, int is);
void labsoc(int i1, int i2, int i3, int i4, int i5, int i6);
void labeoc(int i1, int i2, int i3, int i4, int i5, int i6);
void labsoi(int i1, int i2, int i3, int i4, int i5, int i6);
void labeoi(int i1, int i2, int i3, int i4, int i5, int i6);
void labpre(int i1, int i2, int i3, int i4, int i5, int i6);
void update_times(void);
void verify_date(int y, int m, int d);
void ortecdate(char *ch, int *y,int *m, int *d);
void ortectime(char *ch, int *h,int *m, int *s);
void dateortec(char *ch, int *y,int *m, int *d);
void timeortec(char *ch, int *h,int *m, int *s);
```

Appendix G - Management of the exercises

Organization and management of this exercise:

Mika Nikkinen
Doletum Oy
Länsiportti 1C16
02210 Espoo
Finland
Tel. +358 40 5212 967
Fax. +358 9 88155600
e-mail: doletum@kolumbus.fi

Sub-project leadership of NKS/BOK-1.1, maintenance of the BOK-1.1 web site:

Sigurður Emil Pálsson
Geislavarnir ríkisins
Rauðarárstíg 10
IS-150 Reykjavík
Iceland
Tel +354 552 8200
Fax +354 552 8202
e-mail: sep@gr.is

Bibliographic Data Sheet **NKS-43**

Title	The Use of Synthetic Spectra to Test the Preparedness to Evaluate and Analyze Complex Gamma Spectra
Author(s)	Mika Nikkinen
Affiliation(s)	Doletum Oy, Finland
ISBN	87-7893-096-0
Date	October 2001
Project	NKS/BOK-1.1
No. of pages	77
No. of tables	2
No. of illustrations	20
No. of references	20

Abstract

This is the report of two exercises that were run under the NKS BOK-1.1 sub-project. In these exercises synthetic gamma spectra were developed to exercise the analysis of difficult spectra typically seen after a severe nuclear accident. The spectra were analyzed twice; first, participants were given short time to give results to resemble an actual emergency preparedness situation, then a longer period of time was allowed to tune the laboratory analysis results for quality assurance purposes. The exercise did prove that it is possible to move measurement data from one laboratory to another if second opinion of the analysis is needed. It was also felt that this kind of exercise would enhance the experience the laboratories have in analyzing accident data. Participants expressed the need for additional exercises of this type, this is inexpensive and an easy way to exercise quick emergency response situations not normally seen in daily laboratory routines.

Key words

Gamma spectrum analysis; Quality assurance; Emergency preparedness; Synthetic data; Nuclear accident

Available on request from the NKS Secretariat, P.O.Box 30, DK-4000 Roskilde, Denmark.
Phone (+45) 4677 4045, fax (+45) 4677 4046, e-mail nks@catscience.dk, www.nks.org.