Draft May 8, 2014
Volker Blobel

Constrained Least Squares Fits with APCALC

Least squares based on equality constraints are an alternative to the usual minimization of χ^2-expressions. Constrained fitting is more general and better suited for minimization with the numerical calculation of first derivatives, and allows in a natural way the use of a full covariance matrix of the data with non-zero non-diagonal elements. The user code is transparent because of the strict separation between model calculation and the χ^2 evaluation. The special solution method used in the package APCALC (of less than 800 lines of code, Fortran 77) allows the use of correlated data, even if the covariance matrix of the data is almost singular. All derivatives necessary for a fast solution are determined by numerical methods. The method can be used, in a simple and transparent way, for data combination (averaging) and fitting, and for Gaussian and non-Gaussian data (log-normal, Poisson data), with systematic uncertainties incorporated in the fit, but also for problems like the error propagation with the transformation of variables.

Contents

1 Constrained least squares **3**

2 Usage **6**
2.1 Preparation of the fit . . . 7
2.2 Fit calculations . . . 8
2.3 Utility subprograms . . . 10
2.4 Comparison of minimization methods . . . 11
2.5 Cpu-time for fit . . . 12

3 Non-Gaussian data **12**
3.1 Poisson data . . . 13
3.2 Log-normal data . . . 14
3.3 Outlier . . . 16

4 Mathematics **16**
4.1 Matrix operations . . . 16
4.2 Numerical differentiation . . . 19
4.3 Work storage . . . 20

5 Use of analytical derivatives **20**

6 Examples **22**
6.1 Pythagorean theorem . . . 22
6.2 Error propagation . . . 23
6.3 Measurement of masses . . . 24
6.4 Peelles problem . . . 25
6.5 Combination of correlated measurements . . . 31
6.6 Histogram fit of a Gaussian peak . . . 34
6.7 Straight line fit for $x - y$-data with uncertainties in both coordinates . . . 35
6.8 ... last but not least . . . 38

Overview over all entries in package APCALC

```
      CALL APC    (NX,X,VX,NF,F,STATUS, ISP,IST,WORK)
      CALL APCALC(INI,NX,X,VX, NF,F,A, DX,STATUS,IP,    WORK)
      CALL APCLIN(INI,NX,X,DX,STATUS, WORK)           ! linear iteration
      CALL APCNON(INI,NX,X,DX,STATUS, WORK)       ! non-linear iteration
      CALL APCPAR(MAXITE,EPSITE)          ! define convergence parameter
      CALL APCPUL(NX,X,VX,PULL, WORK)                          ! get pull
      CALL APCOVA(NX,VX, WORK)                    ! get covariance matrix
      CALL APCRES(NX,X,VX, WORK)! print start X and fitted X with errors
      CALL APCERO(NX,X,VX,NF,F)                                  ! zero
      CALL DCMINV(W,B,NARG, NRANK, AUX,NEXT)          ! matrix inversion
      CALL DXMINV(W,B,NX,NF, NRANK, AUX,NEXT)      ! solve linear system
      CALL APCOPY(X,VX,IX,Y,VY,JY,NN)             ! copy symmetric matrix
      CALL APRXVX(NX,X,VX)                       ! print X, VX with errors
      CALL APCORR(NX,VX)                            ! print correlations
      CALL APDER (NX,NF,A)                       ! print derivative matrix
      CALL PRODER(F,D,H)                      ! internal (num derivatives)
      CALL APCDER(NX,X,NF,F,AJAC,ICON,WORK) ! internal (num derivatives)
      CALL APCEPS(ARG)                              ! multiply by factor
```

Common for APCALC applications: apcom.inc

```
      IMPLICIT NONE
*     apcalc arrays
      INTEGER NXMAX,NFMAX,NMAX,NSYMAX, NX,NF,INI,ICON,IST,ISP
*     ------------------------------
      PARAMETER (NXMAX=5,NFMAX=5)  !          <===< adapt these numbers
      PARAMETER (NMAX=NXMAX+NFMAX,NSYMAX=(NMAX*NMAX+NMAX)/2)
      DOUBLE PRECISION X(NXMAX),VX((NXMAX*NXMAX+NXMAX)/2)
      DOUBLE PRECISION F(NFMAX)
      DOUBLE PRECISION WORK(NSYMAX+5*NMAX+5*NFMAX)
      DOUBLE PRECISION STATUS(6)
```

Model fit program

```
      NX=...
      NF=...
      CALL APCERO(NX,X,VX,NF,F)
      X(1)= ...
      ...
      VX(1)=...
      ...
      ISP  = steering parameter e.g. =1
   10 CONTINUE              ! next iteration
   20 F(1) = ...            ! next calculation of constraints
      ...    ...
      F(NF)= ...
      CALL APC(NX,X,VX,NF,F,STATUS, ISP,IST,WORK)
      IF(IST.GT.0) GOTO 10  ! next iteration
      IF(IST.LT.0) GOTO 20  ! continue derivative calculation
```

1 Constrained least squares

In standard least squares methods the *measured variables*, with measured values y_i, $i = 1, 2 \ldots$ and (unmeasured) *model parameters* $a_j, j = 1, 2 \ldots$ are distinguished. The model $f_i(\boldsymbol{a})$ predicts the measured values: $\mathrm{E}[y_i] = f_i(\boldsymbol{a})$. For any values of the parameters $\boldsymbol{a}$ the residuals, the difference between the measured value y_i and the model prediction $f_i(\boldsymbol{a})$ can be calculated: residual $= y_i - f_i(\boldsymbol{a})$. For the least squares calculation a so-called χ^2-expression is constructed

$$\chi^2(\boldsymbol{a}) = \sum_i \frac{\left(y_i - f_i(\boldsymbol{a})\right)^2}{\sigma_i^2} \tag{1}$$

with the sum of the squared residuals, divided by the variance σ_i^2 of the data, which has to be *constant*! Here it is assumed that the measured values are uncorrelated, i.e. the covariance matrix $\boldsymbol{V}_y$ is diagonal; for correlated measured values (covariance matrix non-diagonal) the inverse covariance matrix $\boldsymbol{V}_x^{-1}$ has to be calculated and used as a weight matrix in a matrix expression, instead of the simple sum of equation (1). The least squares estimate for the parameters, denoted by $\widehat{\boldsymbol{a}}$, is obtained by minimization of the χ^2-expression (1); the minimization requires to solve a system of m linear equations for the case of m parameters. One step is sufficient, to get the least squares solution, if the model prediction is a linear function of the parameters. For non-linear models several iterations are required until convergence and approximate values as starting value for iterations are necessary.

In the χ^2-formalism above residuals are calculated between a *single measured value* and an *expression*, which does not depend on any measured value, but only on the *unmeasured* parameters. This formalism can be applied to many least squares problems, but for certain problems the formalism is difficult to apply or even impossible; this is the case, if more than one measured value is involved in a single equation.

In particle physics measurements quite a number of systematic uncertainties exist and have to be accounted for in statistical calculations. An example for an attempt to include a number of different systematic sources of uncertainties, the χ^2-expression

$$\chi^2_{\mathrm{exp}}(\boldsymbol{m}, \boldsymbol{b}) = \sum_i \frac{\left[m^i \left(1 - \sum_j \gamma_j^i b_j\right) - \mu^i\right]^2}{\delta^2_{i,\mathrm{stat}} \mu^i \left(m^i \left[1 - \sum_j \gamma_j^i b_j\right]\right) + \left(\delta_{i,\mathrm{uncor}} m^i\right)^2} + \sum_j b_j^2 \tag{2}$$

has been constructed to allow the averaging of DIS data from Hera experiments[21], with the statement "*the minimum of equation ... provides an unbiased estimate for* $\boldsymbol{m}$". Here the μ_i are measured values and values m^i are fitted (details are given in this reference, where also further references are given). According to the paper, the χ^2-expression takes into account the multiplicative nature of leading uncertainties and by assuming the statistical uncertainty determined by the square root of event numbers, apparently to assume a Poisson distribution (compare section 3 on treatment of non-Gaussian data). The statistical properties of χ^2-expression (2) (with fitted values in the denominator!), will depend on the used minimization strategy (see remark at the end of section 3.1) and the claimed "*unbiasedness*" is – at least – not easy to retrace. Both a decrease of the nominator and an increase of the denominator will give a decrease of the χ^2-function, because the denominator depends on certain fitted parameters and usually such property will result in biased fit parameters[23]. Note that in the above expression the variance is not a constant and if a Gaussian pdf is assumed for the measured quantities with a non-constant variance, the log-likehood contribution (times -2)

includes a term $+\ln\sigma^2$ depending on the (variable) variance (or standard deviation). This extra term $\ln\sigma^2$ is usually ignored in the χ^2-function minimization and this may result in a biased solution. Furthermore a generalization of the above χ^2-expression to take into account correlated uncertainties (i.e. a non-diagonal covariance matrix for the measured data) seems not to be straightforward. The χ^2-minimization method is described in an earlier publication[22]. According to a Monte Carlo calculation "*purity and stability typically exceed 50 %*", which means that more than 50 % of the events either recorded or generated in a bin are measured in the correct bin; this statement could point to a non-negligible correlation between neighbour bins, which would result in a non-diagonal covariance matrix.

An alternative least squares formalism, which is more general than minimization of χ^2-expressions and which can be applied to more cases, is *least squares with equality constraints*[1]. Here *constraint equations* of the type $F_j(\widehat{\boldsymbol{y}}, \widehat{\boldsymbol{a}}) = 0$ are defined. The mathematical problem is to find corrections $\boldsymbol{\Delta y}$ to the measured values $\boldsymbol{y}$ and corrections $\boldsymbol{\Delta a}$ to the initial values $\boldsymbol{a}$ of the parameters, such that the equality constraints are satisfied:

$$F_j\left(\boldsymbol{y} + \boldsymbol{\Delta y},\, \boldsymbol{a} + \boldsymbol{\Delta a}\right) = F_j\left(\widehat{\boldsymbol{y}},\, \widehat{\boldsymbol{a}}\right) = 0 \qquad j = 1,\, 2 \ldots \,. \tag{3}$$

In addition to the fitted values $\widehat{\boldsymbol{a}} = \boldsymbol{a} + \boldsymbol{\Delta a}$ of the parameters, fitted values $\widehat{\boldsymbol{y}} = \boldsymbol{a} + \boldsymbol{\Delta y}$ of the measured variables are calculated. The corrections $\boldsymbol{\Delta y}$ have to be minimal in the sense that the expression

$$S(\boldsymbol{\Delta y}) = \boldsymbol{\Delta y}^{\mathrm{T}} \boldsymbol{V}_y^{-1} \boldsymbol{\Delta y} \tag{4}$$

has to attain the minimal value; this expression is already written for the general case of a non-diagonal covariance matrix $\boldsymbol{V}_y$ of the data.

In the simple case above, where the residual definition is applicable, one constraint is used for each residual, with

$$F_i\left(\Delta y_i, \boldsymbol{a} + \boldsymbol{\Delta a}\right) = \widehat{y}_i - f_i(\widehat{\boldsymbol{a}}) = \left(y_i + \Delta y_i\right) - f_i\left(\boldsymbol{a} + \boldsymbol{\Delta a}\right) = 0$$

i.e. the equality constraints are defined without a reference for the *uncertainties*, and any *non-linearity* in the model appears in the constraints. The uncertainties are taken into account, without any reference to the model, in the minimized expression (4). Thus there is a strict separaration between the *model assumption* and the *minimization*. The alternative method requires a *different* formal description of the problem, which however appears not to be more complicated than the use of χ^2-expressions (1).

The advantage of the constraint least squares becomes apparent for different and more difficult least squares problems. A simple example is the case, where the masses of two bodies are independently measured (measurements y_1 and y_2), and in addition the total mass of the two bodies is measured (measurement y_3). A single constraint $F_1 = \widehat{y}_1 + \widehat{y}_2 - \widehat{y}_3$ has to be applied in order to get improved values of the two masses. In section 6 this problem is shown as an example, where in addition the small difference of the two masses is measured (measurement y_4) with higher precision, which allows a second constraint $F_2 = \widehat{y}_4 - (\widehat{y}_1 - \widehat{y}_2)$. The result is a higher accuracy for the two masses.

The implementation of constrained least squares in APCALC has certain special features. A full covariance matrix is always assumed, with the corresponding treatment of correlated variables. The

[1] In the paper constraints are always *equality constraints*, which have to be satisfied in the fit within round-off errors of the digital computation. This is different from the use in several particle physics papers, which use *constraint* in a loose sense, e.g. "... we incorporate an extra constraint $\tau = \tau_c \pm \sigma_c$ ...".

mathematical solution does not require the inversion of the original covariance matrix $\boldsymbol{V}_y$, as seems to be indicated by equation (4) (see explanation in section 4.1). This avoids the loss in precision, which occurs in the inversion of matrices close to singular, but reduces in addition the execution time. The minimization uses the first derivatives of the constraints, to obtain the solution for linear problems in a single step. After the fit the full covariance matrix of measured and unmeasured variables is available, which corresponds to the inverse of the second-derivative matrix of the χ^2-expression. By default the derivatives (first derivatives of the constraint equations), necessary in the constraint formalism, are determined in APCALC by numerical methods. The implementation treats measured variables (denoted by $\boldsymbol{y}$ above) and parameters or unmeasured variables (denoted by $\boldsymbol{a}$ above) formally in the same way: both types of variables are denoted by $\boldsymbol{x}$ in the following; the only difference between variables of the two classes from the users point of view is that the elements of the input covariance matrix corresponding to unmeasured variables are zero. The transition of variables from one class to the other is trivial.

The mathematical problem of least squares with constraints, expressed with the notation of $\boldsymbol{x}$ for both measured and unmeasured variables

$$\boxed{\begin{array}{l}\text{minimum of } \boldsymbol{\Delta x}^{\mathrm{T}} \boldsymbol{V}_x^{-1} \boldsymbol{\Delta x} \\ \text{subject to } F_j(\boldsymbol{x}) = 0 \quad \text{for } j = 1, 2 \ldots\end{array}} \tag{5}$$

is solved by the method of Lagrangian multipliers, using a block matrix method, which avoids the inversion of matrix $\boldsymbol{V}_x$. The application area of least squares with constraints, as written in equation (5) (or equations (3) and (4)), includes several different problems classes:

- χ^2 problems with simple residuals, as mentioned above;
- error propagation for transformed variables;
- data combination (averaging) with correlated and uncorrelated systematic uncertainties;
- Gaussian and non-Gaussian variables (Poisson, log-normal)
- true conditions between variables (e.g. $a^2 + b^2 = c^2$ for sides a, b and c in a right-angled triangle)
- regularization for ill-posed problems (Tikhonov [28]) by assigning approximate values and uncertainties to unmeasured variables, equivalent to norm regularization.

In the following the notation with variables x_j for both measured and unmeasured variables is used and written in terms of program variables. Measured and unmeasured variables `X(.)` with covariance matrix `VX(.)` (in symmetric storage mode) are assumed. By convention the rows and columns of `VX(.)` corresponding to unmeasured variables have to be zero. Equality constraints are in the form of expressions $F(X) = 0$. The user has to provide value for all variables including covariance matrix for measured values and approximate values for unmeasured variables

```
NX, X(.), VX(.)
```

and the current values of constraints

```
NF, F(.)
```

Internally the matrix `A(.)` of first derivatives of the constraints with respect to all variables: is calculated for each step, by numerical methods. Corrections to the original values in a user array `DX(.)` of the variables `X(.)` are determined in each step.

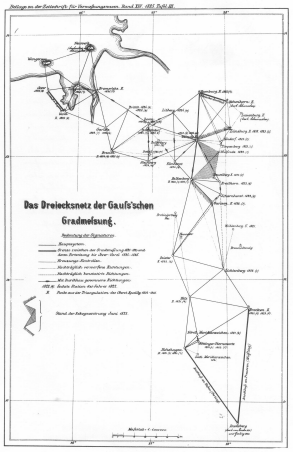

Figure 1: Carl Friedrich Gauss (1777 – 1855). Stone on the summit of mount *Wilseder Berg* (169 m) near Hamburg, commemorating the survey. The Wilseder Berg was an important triangulation station in the great survey in northern Germany between Götttingen and Altona, known as the "Hannoversche Gradmessung" (1821 – 1823).

History. The least squares method with linear and non-linear equality constraints is in use since decades and even centuries. In the Lagrange formalism non-linear constraints require an iterative method for the determination of the stationary point of the Lagrange function, while problems with linear constraints are solved in a single step. Very early applications came mainly from geodesy and surveying[1]. In 1820 King George IV of England, who was also the King of Hannover, tasked the Professor of Astronomy and Director of the Observatory at Göttingen University, Carl Friedrich Gauss, to survey the Kingdom of Hannover, see Figure 1. The least squares method has been used extensively in the geometrical and kinematical analysis of bubble chamber data [2, 3], when digital computers became available around the middle of the 20.th century. Existing constraint fitting programs in high-energy physics are often specialized, e.g. to fits of particle reactions with conservation of energy and momentum, and not easily adapted to different problems. They are used e.g. in the discovery of the top quark.
The program package APCALC is an improved version of a program package CONDFIT, in use at various experiments (PLUTO at DORIS and PETRA, H1 at HERA) since the nineteenhundertseventies. A paper with the titel "*Constrained Least Squares and Error Propagation*", can be found on the authors home page[5] since the year 1997. The algorithmus is described in a text book[6].

2 Usage

The user has to write a short program, where (1) the input data are prepared and (2) the constraint equations are specified. Almost all program variables, which are listed in Table 1, are in user arrays, i.e. the user has to declare the variables and arrays with appropriate dimension parameters. The array WORK(.) is an auxiliary array, used internally for matrix operations and numerical differentiation. The section 6 contains example programs which illustrate the structure of program applying the APCALC package.

	variable	meaning	dimension
N_x	NX	number of variables (measured and unmeasured)	1
$\boldsymbol{x}$	X(.)	array of variables	NX
$\boldsymbol{V}_x$	VX(.)	array for covariance matrix	(NX*NX+NX)/2
		(elements for unmeasured variables zero)	
N_F	NF	number of constraint equations	1
$\boldsymbol{F}$	F(.)	array of costraint values	NF
	STATUS(6)	status of fit	4
	STATUS(1)	current χ^2-value	
	STATUS(2)	previous χ^2-value	
	STATUS(3)	current $\|F\|$ average	
	STATUS(4)	previous $\|F\|$ average	
	STATUS(5)	number of degrees of freedom	
	STATUS(6)	number of evaluations of constraints	
	INI	iteration counter	1
$\boldsymbol{A}$	A(NX,NF)	array for Jacobian matrix (first derivatives)	NX*NF
$\boldsymbol{\Delta x}$	DX(NX)	array of corrections	NX
	IP	print flag	
	PULL(.)	vector of pulls	NX
	WORK(.)	work array	(N*N+N)/2+5*N +5*NF
			with N=NX+NF

Table 1: User variables and variable arrays and information available after the calculation. All real data in double precision.

2.1 Preparation of the fit

In the first part the data for the fit are prepared. The arrays etc. have to be declared; an example, contained in the file apcom.inc is shown below.

```
*     Common for APCALC applications: apcom.inc
*     ------------------------------------------------------------------
      IMPLICIT NONE
*     apcalc arrays
      INTEGER NXMAX,NFMAX,NMAX,NSYMAX,  NX,NF,INI,ICON,IST,ISP
*     ----------------------------
      PARAMETER (NXMAX=5,NFMAX=5)   !            <===< adapt these numbers
      PARAMETER (NMAX=NXMAX+NFMAX,NSYMAX=(NMAX*NMAX+NMAX)/2)
      DOUBLE PRECISION X(NXMAX),VX((NXMAX*NXMAX+NXMAX)/2)
      DOUBLE PRECISION F(NFMAX)
      DOUBLE PRECISION WORK(NSYMAX+5*NMAX+5*NFMAX)
      DOUBLE PRECISION STATUS(6)
*     ------------------------------------------------------------------
```

Here the two parameters NXMAX and NFMAX with the maximum values of the dimension parameters NX and NF have to be given.
The actual values of the two dimension parameters NX and NF have to be declared and the arrays X(.), VX(.), F(.) have to be reset to zero (see subroutine APCERO in section 2.3 on page 10).

```
      NX=...
```

```
NF=...
CALL APCERO(NX,X,VX,NF,F)
```

The arrays X(.) with the measured values and start values for unmeasured values have to be set, and the array VX(.) has to be filled with the elements of the covariance matrix of the measured data.

```
X(1)= ...
...
VX(1)=...
...
```

The convention is to have the values zero for all elements of the array VX(.) corresponding to unmeasured values.

2.2 Fit calculations

The second part includes the actual calculations. The preparation for a minimization step includes the calculation of the constraints with the actual value of the variables X(.). The iterations are steered by the variable ISP with the following meaning:

```
ISP =  0  quiet      3-point derivatives   "almost" linear problem
      +1  printout
      +2  non-linear problem
      +4  5-point derivatives
```

Examples: for a simple problem with linear constraints the value ISP=1 should be used, if printout is required; for a difficult problem with non-linear constraints ISP=7 should be used, if printout is required.

The next part includes the constraint calculation in a loop. At the end of a loop the corrections to the variables are calculated and the variables are updated.

```
   ISP  = steering parameter
10 CONTINUE              ! next iteration
20 F(1) = ...            ! next calculation of constraints
   ...     ...
   F(NF)= ...
   CALL APC(NX,X,VX,NF,F,STATUS, ISP,IST,WORK)
   IF(IST.GT.0) GOTO 10  ! next iteration
   IF(IST.LT.0) GOTO 20  ! continue derivative calculation
```

First derivatives of all constraints are calculated internally; the variables are modified and the constraints are re-calculated. The flag IST controls the loop; the loop is left for the flag value IST=0 after convergence. The code above allows to modify the input covariance matrix for each new iteration; this possibility, meaningful e.g. for Poisson-distributed variables, corresponds to the *Iteratively Reweighted Least Squares* (IRLS)[2] method.

Convergence. After convergence the fitted values of the variables are available in the array X(.). Initial values and fitted values, uncertainties and pulls can be printed by

```
CALL APCRES(NX,X,VX, WORK) ! print result
```

[2]IRLS is used to find the maximum likelihood estimates of a generalized linear model, and in robust regression to find an M-estimator. One of the advantages of IRLS over linear and convex programming is that it can be used with Gauss-Newton and Levenberg-Marquardt numerical algorithms (from WIKIPEDIA).

After this call the fitted covariance matrix and the pulls can be obtained by the following calls, from the arrays DX(.) and WORK(.) used in the fit, in the given order (the content of the covariance array VX(.) is replaced in the call of APCOVA).

```
CALL APCPUL(NX,X,VX,PULL, WORK)   ! get pulls PULL(.)
CALL APCOVA(NX,VX,        WORK)   ! get covariance matrix VX(.)
```

The final values of χ^2 and of $\sum_j |F_j|/\mathrm{NF}$ are available in the array STATUS, and the number of degrees of freedom of the fit in NDF. After the call of APCOVA the elements of the matrix VX(.) are non-zero too for the unmeasured values, which were determined in the constrained fit. The matrix contains all variances and covariances of measured and unmeasured variables after the fit. A subset of variables and their covariance matrix can be obtained (copied) from the full arrays by a call of APCOPY (see section 2.3.

Pulls. After a fit the statistical indicators like the χ^2-value with number n_{df} of degree of freedom should be checked, to investigate wether the input data values and their uncertainties are statistically compatible with the assumed model. The p-value, calculated from the χ^2-value and n_{df}, should be neither close to 0 nor to 1. Often the scaled residual, i.e. the difference *measured* minus *fitted* value of a variable, divided by the standard deviation of the measured value, is calculated, and it is naively expected, that the distribution of this scaled residual is close to the standardized normal distribution $N(0,1)$. However even under optimal conditions the scaled residual is systematically narrower, because of the influence of the measured value on the fitted value. A better quantity is the quantity *pull*[6, 29], which takes into account the correlation between the errors of the measured and fitted values. The pull is the ratio of the difference *measured* minus *fitted* value, divided by the standard deviation of the difference $\sigma = \sqrt{V_{mm} - V_{ff}}$; here the covariance matrix elements V_{mm} and V_{ff} are the variances of the variable before the fit and after the fit. The distribution of the pull should follow the standardized normal distribution $N(0,1)$. In APCALC pulls for all measured variables are available.
If one variable out of many in a fit is strongly biased or has a larger uncertainty than expected, one could naively expect a large pull of this variable. Unfortunately this is not the case, because the total χ^2 is minimized, and thus a discrepancy is distributed over all variables. Nevertheless the analysis of pulls can give information on the data quality, especially if a large number of data sets is available.

Precision aspects. The calculation of the derivatives of the constraints by numerical methods simplifies the use of the program. However the numerical calculations of derivatives require a high accuracy for the constraint calculation.

> **All APCALC -variables are in double precision. The complete computation of the constraints should be done in *double precision*; otherwise the numerical derivatives are inaccurate and convergence may be impossible.**

The step size for the numerical calculation of derivatives is determined internally, assuming a "realistic" (practical) value of $\varepsilon = 7 \cdot 10^{-17}$ for the upper bound for the double-precision floating point representation of numbers (instead of the nominal value of $6 \cdot 10^{-16}$). The default value corresponds to a constraint calculation in (full) double precision. If it is done partly in single-precision, a larger value for ε should be used. This is done by the call

```
CALL APCEPS(DFACT)                ! double precision factor
```

which modifies the standard value by multiplication with the given factor DFACT by ε = DFACT $\times 7 \cdot 10^{-17}$.

Convergence recognition. Two numerical values are used for convergence recognition and for the limitation of the number of iterations. These number are:

```
MAXITE = 10              ! maximum number of iterations
EPSITE = 1.0 D-8         ! double-precision convergence constant
```

Convergence is assumed if the χ^2-difference in the last two iterations is less than EPSITE and the constraint accuracy $\sum_j |F_j|/\mathrm{NF}$ is less than EPSITE too. Eventual constraints have to be scaled up or down, if they have a different magnitude. The two numbers can be specified by the call:

```
CALL APCPAR(MAXITE,EPSITE)     ! redefine numbers
```

The changed values remain valid until a further change.

2.3 Utility subprograms

Four utility subprograms are part of the APCALC -package, although they are general and not directly related to the minimization package. In the symmetric storage mode only half the matrix elements including the diagonal elements are stored. The storage order is

$$A_{11} \quad A_{12} \quad A_{22} \quad A_{13} \quad A_{23} \quad A_{33} \quad A_{14} \quad \ldots \quad A_{nn}$$

for a n-by-n symmetric matrix, which has in total $(n^2+n)/2$ different elements. The linear index ℓ of element A_{ij} is given by

$$\begin{aligned} \ell &= i + (j^2 - j)/2 && \text{if } i \le j \\ \ell &= j + (i^2 - i)/2 && \text{if } i \ge j\ . \end{aligned}$$

Copy symmetric matrix-part. A symmetric matrix or a submatrix of a symmetric matrix is copied to another symmetric matrix, in symmetric storage mode.

```
CALL APCOPY(X,VX,IX,Y,VY,IY,N)
```

In the APCOPY call N rows and columns of the matrix array VX, starting from the IX-th diagonal element are copied to the matrix array VY, starting from the IY-th diagonal element. Thus if a complete symmetric matrix has to be copied IX=IY=1 has to be used. The N elements of the vector X, starting at element IX, are copied too to the vector Y, starting at element IY.

Print fit result. The NX elements of the vector X are printed , together with square root of the corresponding diagonal elements of the covariance matrix VX.

```
CALL APRXVX(NX,X,VX) ! print X, VX with errors
```

An example for a printout is:

```
par_i        X_i_____+-std.dev.
    1     3.09379       0.951857E-01
    2     4.06734       0.118381
    3     5.11026       0.862333E-01
```

Print correlations. The correlation coefficients ρ_{ij} from the covariance matrix VX are printed.

```
CALL APCORR(NX,VX)    ! print correlations
```

Correlation coefficients by definition have the range $-1 \leq \rho_{ij} \leq +1$. An example for a printout is:

```
Correlation coefficients between parameters i and j:
par_i  \ j=     1      2      3
    1        1.000
    2       -0.439  1.000
    3        0.189  0.800  1.000
```

Print derivative matrix. This printout is useful, if *analytic* derivatives are used and the correctness of the code has to be checked, by comparison of the values with the numerical derivatives. The NX-by-NF Jacobian $\boldsymbol{A} \equiv$ A(NX,NF) is printed.

```
CALL APDER(NX,NF,A)   ! print derivative matrix
```

2.4 Comparison of minimization methods

The method of constrained fitting is a *more general* least square method than the usual method for the minimization of a χ^2 function. In the special case where the constraints are simple residuals, i.e. the difference between a single measured value and an expectation, given by a parametrization, a χ^2-function corresponds to the sum of terms $F_j(\boldsymbol{x})/\sigma_j^2$, where σ_j^2 is the variance of the measured value. In the χ^2 minimization case there is a single function $\chi^2(\boldsymbol{x})$ and the first and second derivatives w.r.t. to the parameters are needed for a fast solution. In the minimization program MINUIT[4] with MIGRAD-option the first derivatives of the function $\chi^2(\boldsymbol{x})$ are calculated; the estimate of the matrix of second derivatives requires several iterations, even for linear least squares problems.
Disadvantages and advantages of the minimization with constraints are:

− The method with constraints uses more fit variables: NX+NF variables, compared to only parameters (unmeasured variables) in the χ^2-minimization case; consequently the matrices are larger, which will take more Cpu-time in the calculation (see notes on Cpu-time below);

\+ no extra code is necessary for a full covariance matrix with correlations between the measured variables;

\+ the first derivatives of all constraint equations allow already the complete solution of in a single step in the linear constraint case;

\+ the numerical first derivative of a single constraint is more accurate and less sensitive to round-off errors, compared to the numerical derivatives of a total sum of squares;

\+ fitted values with uncertainties (and full covariance matrix) are calculated not only for the parameters, but for all measured values too;

\+ pulls for each measured variable are available after the constrained fit without any extra calculation; note that the pull is different from the normalized residual;

\+ true constraints (e.g. sum of angles in a triangle equal to π) can be included in the constrained fit;

+ constraints can depend on more than one measured value; in χ^2-minimization this requires a special construction of χ^2 contributions;

+ non-Gaussian measured variables like log-normal or Poisson data are easily included in constrained fits;

+ unmeasured and measured variables are treated by the same formalism in the constrained case, making the transition from one class to the other class trivial, e.g. assumption of a *measured value* for a parameter (regularisation), or moving a measured variable to the class of unmeasured variables.

2.5 Cpu-time for fit

As mentioned above, APCALC has a higher Cpu-time consumption compared to χ^2-minimization, if both methods are applicable. The Cpu-time has been measured in an example of a 100-bin histogram of a Gaussian peak with constant background, which can be fitted by χ^2-minimization with four parameters. In APCALC there are 104 variables. For good/bad start values of the fit parameters the time of 30 / 60 msec is used by APCALC (on a Core 2 Duo 2.4 GHz PC), while a method like MINUIT (with numerical determination of the first derivatives) is an order of magnitude faster, and a method like LVMINI (with analytical determination of derivatives) is several orders of magnitude faster. Thus APCALC is slower, on the other hand, a full covariance matrix of the data was assumed, and all 100 pulls, the full covariance matrix of all $4 + 100$ fitted variables and the fitted curve is already calculated. The time is about 10 % higher, if the data values are assumed to be Poisson-distributed (see next section on Non-Gaussian data). Of course the main application area of APCALC are problems, which can not be solved by χ^2-minimization or where the construction of the χ^2-expresssion is difficult or unclear, or where measured values have a full covariance matrix. In the case of a full covariance matrix (with non-zero covariances) of the data the normal χ^2-minimization would take more time.

In another test the time for averaging $n = 100$ and $n = 1000$ values, measured in two experiments, is determined. In addition to the $2 \times n$ measured variables the n averages belong to the variables. There are $2 \times n$ constraint equations. The data sets of each n data points for the two experiments have a full covariance matrix (all n^2 elements non-zero). The solution is found in two iterations and the Cpu-time for $n = 100$ is $T_{\mathrm{CPU}} = 0.35$ sec, and the Cpu-time for $n = 1000$, i.e. in total 3000 variables, is $T_{\mathrm{CPU}} =$ 9 min. The time will of course increase, if non-linearities and additional systematic uncertainties have to be included.

3 Non-Gaussian data

Underlying the standard least squares formalism is the assumption of *fixed variances* of the data, i.e. the variance is *independent* of the fitted value of the variables. A basic theorem for linear least squares about the optimality of linear least squares requires *unbiased* data with fixed variance, while a "Gaussian" distribution is not necessarily required.
Data in applications can have variances which depend on the mean values. A consequence of this fact is an *asymmetric* probability density. The examples most often encountered are *counted data*, where the measured value is a non-negative integer, which follows a *Poisson distribution* (variance $\propto$ mean), e.g. a histogram bin content. Another example are *factors*, which follow the asymmetric

log-normal distribution ($\sqrt{\text{variance}} \propto$ mean). If the dependence of the variance on the mean and the asymmetry of the probability density is ignored, by applying a standard least squares fit, the fit results will have a bias.
Deviations of data properties from the standard Gaussian distribution can be accounted for in APCALC by the IRLS and by a data transformation. The asymmetry of the probability densities is accounted for in APCALC by a *non-linearity* of the constraints, while the χ^2-calculation remains unchanged and simple. Complicated modifications of the χ^2-expressions to account for some asymmetry are avoided. The *iteratively reweighted least squares* (IRLS) method can be used e.g. for Poisson data. Further applications which can be solved with APCALC are maximum likehood estimates of the generalized linear model[24, 25], and in robust regression to determine an M-estimator[26], to reduce the influence of outliers in an otherwise Gaussian-distributed data set.

3.1 Poisson data

Counts are non-negative integer data that follow the Poisson distribution, with variance proportional to the mean value:

$$\mathrm{Var}[x] \propto \mathrm{E}[x] \; .$$

The often used (symmetric) Gaussian approximation of the asymmetric Poisson distribution with variance equal to the *observed* count will result in a small bias, especially for small values of the count x. Even more important, ignoring e.g. bins with zero observed count (which is done in certain standard software) would result in a systematic bias of regions of low probability. The characteristic statistical properties of Poisson data can be taken into account by a redefinition (IRLS) of the variance before each new step. *During the step* the variance has to be *kept fixed*, to avoid a bias in the fit[23]. This technique is applied in the small example below; another example is given in section 6.6, where the content of histogram bins is treated by the IRLS technique.

Averaging Poisson data

The data are two counted numbers x_1 and x_2, which follow the Poisson distribution. Both numbers are assumed to be independent measurements of the same signal,

$$x_1 = 9 \pm \sqrt{9}$$
$$x_2 = 16 \pm \sqrt{16}$$

and the averaging constraint is $F_1(x) = x_1 - x_2$. Averaging with *fixed* variance for the two data would give an average below 12.5, because the value x_1 with the smaller variance would get a larger weight. In the code the variance of the two data is, at the start of each iteration, set to the value fitted so far, as is true for a Poisson variable. The relevant lines of code are:

```
      NX=2
      NF=1
      CALL APCERO(NX,X,VX,NF,F)
      X(1)= 9.0D0
      X(2)=16.0D0
10    VX(1)=X(1)                    ! reweighting
      VX(3)=X(2)                    ! reweigthing
20    F(1)=X(1)-X(2)                ! <-- constraint
      CALL APC(NX,X,VX,NF,F,STATUS,  1,IST,WORK)
```

```
      IF(IST.GT.0) GOTO 10          ! next iteration
      IF(IST.LT.0) GOTO 20          ! continue derivative calculation
      CALL APCRES(NX,X,VX, WORK)           ! print result
```

and the printout from the program is:

```
 iter   ndf       chi^2_____delta          |F|/NF_________delta       ratio
   1      1       1.960                     7.000
   2      1       2.127     0.1667          0.000         -7.000      -8.85
   3      1       1.960    -0.1667          0.000          0.000       0.00
   4      1       1.960     0.000           0.000          0.000       0.00
        20 F-evals (ISP=1, 3-point num.diff. calculation)

 par_i  fitted X_i_____+-std.dev.      initial X_i_____+-std.dev       pull
     1     12.5000        2.50000          9.00000        3.53553      1.40
     2     12.5000        2.50000          16.0000        3.53553     -1.40
```

The fitted result for the average is $25/2 = 12.5$, because of the adjustment of the variances of the two measured values before each iteration at statement number 10, according to IRLS.

3.2 Log-normal data

Certain continuous variables can take only positive values. Often the distribution of these variables can be described by the log-normal distribution. The multiplicative central limit theorem suggests a log-normal distribution for variables, which are the *product* of many independent positive random variables. Examples for log-normal data are normalization factors and systematic multiplicative uncertainties. The often seen publication of a measured value together with a *relative* uncertainty (e.g. given in %) indicates a log-normal distribution of the measured value, which has to be taken properly into account. The log-normal distributiuon is asymmetric; log-normal random variables have the interesting property, that the inverse $1/x$ of a variable x follows the log-normal distribution too. In particle physics a standard calculation is the division of an observed count of the number of events (Poisson) by the integrated luminosity (log-normal), thus two asymetric probability distributions are involved in one operation.

If x has a normal distribution with mean μ and variance σ, then $z = \exp(x)$ has a log-normal distribution. The *relative* standard deviation of a variable z, following the log-normal distribution is *constant*.
The probability density function of the log-normal distribution is

$$f(z;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma}\frac{1}{z}\exp\left[-\frac{(\ln z-\mu)^2}{2\sigma^2}\right] \tag{6}$$

and mean and variance of the factor z can be expressed by the two parameters μ and σ:

$$\text{mean } \mathrm{E}[z] = \exp\left(\mu+\frac{1}{2}\sigma^2\right) \tag{7}$$

$$\text{variance } \mathrm{Var}[z] = \exp\left(2\mu+\sigma^2\right)\left[\exp\left(\sigma^2\right)-1\right]\ . \tag{8}$$

The *relative* standard deviation of the factor z is

$$\varepsilon_{\text{rel}} = \frac{\sqrt{\mathrm{Var}[z]}}{\mathrm{E}[z]} = \sqrt{\exp\left(\sigma^2\right)-1}\,. \tag{9}$$

For a given mean value $\overline{z}$ and *relative* standard deviation ε_{rel} of the log-normal distributed factor z the parameter μ and σ of the Gaussian are given by

$$\mu = \ln \overline{z} - \frac{1}{2}\sigma^2 \qquad\qquad \sigma = \sqrt{\ln\left(1 + \varepsilon_{\text{rel}}^2\right)} \,.$$

In the following a mean value $\overline{z} = 1$ (i.e. $\mu = -\frac{1}{2}\sigma^2$) is assumed with an relative uncertainty ε_{rel}. The maximum of the probability density is, for typical values of ε_{rel}, only slightly shifted to a value below 1, and the value of σ is close to ε_{rel}. The normalization factor with log-normal distribution can be parametrized by a variable x with measured value 0 and standard deviation $\sigma = \varepsilon$ and be expressed by factor $\cdot \exp(x)$ with factor equal to 1. The table

ε_{rel}	factor	σ
1 %	1.0000	0.0100
10 %	0.9950	0.0998
20 %	0.9615	0.1980
50 %	0.8944	0.4724

for relative uncertainties up to 50 % shows that the above approximations are in general accurate enough, compared to the often limited knowledge on the properties of the normalization factors. The expression for the normalization can never become zero or negative in a fit.

An example for the usual approximate treatment of normalization uncertainties in χ^2-expressions is given in a publication on fits of parton distribution functions with data from many experiments:

> "... Then, in addition, the fully correlated normalization error of the experiment is usually specified separately. For this reason, it is natural to adopt the following definition for the effective χ^2 (as done in previous CTEQ analyses):
>
> $$\chi^2_{\text{global}}(a) = \sum_n w_n \chi^2_n(a) \qquad (n \text{ labels the different experiments}), \tag{2}$$
>
> $$\chi^2_n(a) = \left(\frac{1 - \mathcal{N}_n}{\sigma_n^N}\right)^2 + \sum_I \left(\frac{\mathcal{N}_n D_{nI} - T_{nI}(a)}{\sigma_{nI}^D}\right)^2 \,. \tag{3}$$
>
> For the nth experiment, D_{nl}, σ_{nl}^D, and $T_{nl}(a)$ denote the data value, measurement uncertainty (statistical and systematic combined) and theoretical values (dependent on $\{a\}$) for the Ith data point, σ_n^N is the experimental normalization uncertainty, and $\mathcal{N}_n$ is an overall normalization factor (with default value 1) for the data of experiment n. ..." (from Ref.[27])

Here a single residual is calculated from two measured values, D_{nI} and $\mathcal{N}_n$. The first term in equation (3) is a symmetric Gaussian expression about the measured value 1, which approximates the unsymmetric log-normal distribution of $\mathcal{N}_n$. The second term with a sum on all residuals from the nth experiment violates the Gaussian rule of constant uncertainty, because the measured value D_{nI} is multiplied by the factor $\mathcal{N}_n$; this construction favors a fitted factor $\mathcal{N}_n < 1$, and generates a bias towards smaller fitted values $T_{nI}(a)$. Such a "*correction*" of the data, called *natural* in the quoted text, is done rather often. It would be better to use the nominator $D_{nI} - \mathcal{N}_n T_{nI}(a)$, which leaves the measured values (and its uncertainty) unchanged and modifies the expectation. The bias due to the

construction as in equation (3) will be often rather small and perhaps negligible, but under certain constellations it can have a large effect; such an example is treated in section 6.4.
With APCALC the fully correlated normalization error is treated in the following way: the constraint equation for a data point is

$$F_I = D_{nI} - \exp(x_n)\, T_{nI}(a)$$

where x_n is measured with $0 \pm \varepsilon_n$, and the element of the covariance matrix corresponding to D_{nI} is equal to $\left(\sigma^D_{nI}\right)^2$ and the element corresponding to x_n is ε_n^2.

The use of a possible weighting factor w_n in equation (2) from Ref.[27] is not commented here; any value of w_n (except the trivial values 0 and 1) would introduce a scale factor for *all* uncertainties of data and fitted parameters from the nth experiment.

3.3 Outlier

In least square fits the influence of data with very large residuals on the result is large, resulting in a bias of fitted parameters. The method of M-estimates[26] is one method, which is robust against outliers, and stable w.r.t small deviations within an assumed parametric model. In this method the weights of data with large residuals are reduced in a generalized maximum-likelihood with a IRLS method; M-estimators are asymptotically normally distributed.

4 Mathematics

4.1 Matrix operations

The solution on APCALC for the correction vector $\boldsymbol{\Delta x}$ for one step requires the solution of a large symmetric matrix equation with NX+NF rows/columns. The solution of this equation with its large matrix, using a block matrix method, which avoids the inversion of matrix $\boldsymbol{V}_x$, is discussed below. In the solution one has to distinguish between the variables, which are *measured* and have non-zero values of the corresponding elements of the covariance matrix, and the variables, which are *unmeasured*. In the formulae given below the vector $\boldsymbol{x}$ and the Jacobian $\boldsymbol{A}$ are partitioned into the measured part $\boldsymbol{x}_m$, with index m, and the unmeasured part $\boldsymbol{x}_u$, with index u. The program code allows any order of m- and u-variables, only in this mathematical description the two types of variables are separated.

The least squares problem can be expressed by

$$\text{variables } = \boldsymbol{x} = \begin{pmatrix} \boldsymbol{x}_m \\ \boldsymbol{x}_u \end{pmatrix} \qquad \text{minimize} \quad S(\boldsymbol{\Delta x}_m) = \boldsymbol{\Delta x}_m^{\mathrm{T}}\, \boldsymbol{V}_x^{-1}\, \boldsymbol{\Delta x}_m$$

$$\text{subject to equality constraints} \quad F_k\left(\boldsymbol{x}_m + \boldsymbol{\Delta x}_m, \boldsymbol{x}_u + \boldsymbol{\Delta x}_u\right) = 0 \qquad k = 1,\, 2 \ldots m\ .$$

The starting values of the variables $\boldsymbol{x}_m$ and $\boldsymbol{x}_u$ will not exactly fulfil the constraints, and *corrections* $\boldsymbol{\Delta x}_m$ to the *measured* variables and $\boldsymbol{\Delta x}_u$ to the *unmeasured* variables have to be determined. If the constraints are linear, the solution is determined in one step. In case of nonlinear conditions the solution of the problem is reduced to a sequence of linear problems by linearization of the conditions. Linearization requires starting values for the variables. For the measured variables, the measurement itself is taken as starting approximation. For unmeasured parameters starting values have to be determined by methods depending on the specific problem (usually selected constraints are used).

A general technique for the determination of local extrema of a function of several variables subject to constraints is the method of **Lagrangian multipliers**, which has the advantage of a completely symmetric treatment of all variables. In this method, m additional parameters λ_k, the Lagrange multipliers, are introduced, one for each equation. Formally a new function is defined:

$$\mathcal{L}(\Delta \boldsymbol{x}_m, \Delta \boldsymbol{x}_u, \boldsymbol{\lambda}) = S(\Delta \boldsymbol{x}_m) + 2 \sum_{k=1}^{m} \lambda_k F_k(\Delta \boldsymbol{x}_m, \Delta \boldsymbol{x}_u) \ , \tag{10}$$

where the factor 2 has been introduced for later convenience. The necessary condition for a stationary point, with zero derivatives of $\mathcal{L}(\Delta \boldsymbol{x}_m, \Delta \boldsymbol{x}_u, \boldsymbol{\lambda})$ with respect to all components of $\boldsymbol{x}_m$, $\boldsymbol{x}_u$ and $\boldsymbol{\lambda}$ is then equivalent to the necessary condition of a minimum of $S(\boldsymbol{x}_m)$ under conditions $F_k(\Delta \boldsymbol{x}_m, \Delta \boldsymbol{x}_u) = 0$.

The linearization is expressed in each iteration in terms of corrections $\Delta \boldsymbol{x}_m$ and $\Delta \boldsymbol{x}_u$ to the starting values $\boldsymbol{x}_m$ and $\boldsymbol{x}_u$. Corrections obtained in the previous step are denoted by $\Delta \boldsymbol{x}_m^*$ and $\Delta \boldsymbol{x}_u^*$. The linearized conditions are

$$F_k(\boldsymbol{x}_m^*, \boldsymbol{x}_u^*) + \sum_j \frac{\partial F_k}{\partial x_i} (\Delta x_i - \Delta x_i^*) \approx 0 \tag{11}$$

where the function values and all derivatives are evaluated at $\boldsymbol{x}_m^* = \boldsymbol{x}_m + \Delta \boldsymbol{x}_m^*$ and $\boldsymbol{x}_u^* = \boldsymbol{x}_u + \Delta \boldsymbol{x}_u^*$ (for the first iteration $\Delta \boldsymbol{x}_m^* = 0$ and $\Delta \boldsymbol{x}_u^* = 0$). In vector notation the set of linearized conditions is

$$\boldsymbol{F} + \boldsymbol{A}_m (\Delta \boldsymbol{x}_m - \Delta \boldsymbol{x}_m^*) + \boldsymbol{A}_u (\Delta \boldsymbol{x}_u - \Delta \boldsymbol{x}_u^*) = 0 \tag{12}$$

or

$$\boldsymbol{A}_m \Delta \boldsymbol{x}_m + \boldsymbol{A}_u \Delta \boldsymbol{x}_u = \boldsymbol{c} \qquad \text{with} \qquad \boldsymbol{c} = \boldsymbol{A}_m \Delta \boldsymbol{x}_m^* + \boldsymbol{A}_u \Delta \boldsymbol{x}_u^* - \boldsymbol{F} \tag{13}$$

The combined matrix $\boldsymbol{A}$ of derivatives, and the vector $\boldsymbol{F}$ of function values are defined by

$$\boldsymbol{A} = \begin{pmatrix} \partial F_1/\partial x_1 & \partial F_1/\partial x_2 & \dots & \partial F_1/\partial x_p \\ \partial F_2/\partial x_1 & \partial F_2/\partial x_2 & \dots & \partial F_2/\partial x_p \\ & & & \\ \dots & & & \\ \partial F_m/\partial x_1 & \partial F_m/\partial x_2 & \dots & \partial F_m/\partial x_p \end{pmatrix} \qquad \boldsymbol{F} = \begin{pmatrix} F_1(\boldsymbol{x}^*, \boldsymbol{y}^*) \\ F_2(\boldsymbol{x}_m^*, \boldsymbol{x}_m^*) \\ \\ \dots \\ F_m(\boldsymbol{x}_m^*, \boldsymbol{x}_u^*) \end{pmatrix} \tag{14}$$

The Lagrange function $\mathcal{L}$ can now be written as

$$\mathcal{L}(\Delta \boldsymbol{x}_m, \Delta \boldsymbol{x}_u, \boldsymbol{\lambda}) = \Delta \boldsymbol{x}_m^{\mathrm{T}} \boldsymbol{V}_x^{-1} \Delta \boldsymbol{x}_m + 2 \boldsymbol{\lambda}^{\mathrm{T}} (\boldsymbol{A}_m \Delta \boldsymbol{x}_m + \boldsymbol{A}_u \Delta \boldsymbol{x}_u - \boldsymbol{c}) \tag{15}$$

The necessary conditions for a stationary point are obtained by differentiation:

$$\begin{array}{lllll} \boldsymbol{V}_x^{-1} \Delta \boldsymbol{x}_m & + & \boldsymbol{A}_m^{\mathrm{T}} \boldsymbol{\lambda} & = & 0 \\ & & \boldsymbol{A}_u^{\mathrm{T}} \boldsymbol{\lambda} & = & 0 \\ \boldsymbol{A}_m \Delta \boldsymbol{x}_m & + \quad \boldsymbol{A}_u \Delta \boldsymbol{x}_u & & = & \boldsymbol{c} \end{array} \ . \tag{16}$$

This system of coupled matrix equations, in block matrix form

$$\begin{pmatrix} \boldsymbol{V}_x^{-1} & 0 & \boldsymbol{A}_m^{\mathrm{T}} \\ 0 & 0 & \boldsymbol{A}_u^{\mathrm{T}} \\ \boldsymbol{A}_m & \boldsymbol{A}_u & 0 \end{pmatrix} \begin{pmatrix} \Delta \boldsymbol{x}_m \\ \Delta \boldsymbol{x}_u \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \boldsymbol{c} \end{pmatrix} \ , \tag{17}$$

has to be solved for the unknowns $\Delta \boldsymbol{x}_m$, $\Delta \boldsymbol{x}_u$ and $\boldsymbol{\lambda}$. The inverse matrix with partitioning is written with the same partitioning:

$$\begin{pmatrix} \boldsymbol{V}_x^{-1} & 0 & \boldsymbol{A}_m^{\mathrm{T}} \\ 0 & 0 & \boldsymbol{A}_u^{\mathrm{T}} \\ \boldsymbol{A}_m & \boldsymbol{A}_u & 0 \end{pmatrix}^{-1} = \begin{pmatrix} \boldsymbol{C}_{11} & \boldsymbol{C}_{21}^{\mathrm{T}} & \boldsymbol{C}_{31}^{\mathrm{T}} \\ \boldsymbol{C}_{21} & \boldsymbol{C}_{22} & \boldsymbol{C}_{32}^{\mathrm{T}} \\ \boldsymbol{C}_{31} & \boldsymbol{C}_{32} & \boldsymbol{C}_{33} \end{pmatrix} . \tag{18}$$

Below formulae are given for each of the submatrices. The abbreviations

$$\boldsymbol{W}_m = (\boldsymbol{A}_m \boldsymbol{V}_x \boldsymbol{A}_m^{\mathrm{T}})^{-1} \tag{19}$$

$$\boldsymbol{W}_u = (\boldsymbol{A}_u^{\mathrm{T}} \boldsymbol{W}_m \boldsymbol{A}_u) \tag{20}$$

are introduced to simplify the equations. The elements $\boldsymbol{C}_{11} \ldots \boldsymbol{C}_{33}$ are

$$\begin{aligned} \boldsymbol{C}_{11} &= \boldsymbol{V}_x - \boldsymbol{V}_x \boldsymbol{A}_m^{\mathrm{T}} \boldsymbol{W}_m \boldsymbol{A}_m \boldsymbol{V}_x \\ &\quad + \boldsymbol{V}_x \boldsymbol{A}_m^{\mathrm{T}} \boldsymbol{W}_m \boldsymbol{A}_u \boldsymbol{W}_u^{-1} \boldsymbol{A}_u^{\mathrm{T}} \boldsymbol{W}_m \boldsymbol{A}_m \boldsymbol{V}_x \\ \boldsymbol{C}_{21} &= -\boldsymbol{W}_u^{-1} \boldsymbol{A}_u^{\mathrm{T}} \boldsymbol{W}_m \boldsymbol{A}_m \boldsymbol{V}_x \\ \boldsymbol{C}_{22} &= \boldsymbol{W}_u^{-1} \\ \boldsymbol{C}_{31} &= \boldsymbol{W}_m \boldsymbol{A}_m \boldsymbol{V}_x - \boldsymbol{W}_m \boldsymbol{A}_u \boldsymbol{W}_u^{-1} \boldsymbol{A}_u^{\mathrm{T}} \boldsymbol{W}_m \boldsymbol{A}_m \boldsymbol{V}_x \\ \boldsymbol{C}_{32} &= \boldsymbol{W}_m \boldsymbol{A}_u \boldsymbol{W}_u^{-1} \\ \boldsymbol{C}_{33} &= -\boldsymbol{W}_m + \boldsymbol{W}_m \boldsymbol{A}_u \boldsymbol{W}_u^{-1} \boldsymbol{A}_u^{\mathrm{T}} \boldsymbol{W}_m \, . \end{aligned} \tag{21}$$

These equations show that the inverse of the covariance matrix $\boldsymbol{V}_x$, which could be called a *weight matrix*, does not appear and has never to be calculated. Only the covariance matrix $\boldsymbol{V}_x$ appears in the formulae. The whole opereration requires certain matrix products and the inversion of two smaller matrices, of dimension N_F-by-N_F (matrix $\boldsymbol{W}_m$), and of dimension N_u-by-N_u (matrix $\boldsymbol{W}_u$).
The calculation of the corrections $\Delta \boldsymbol{x}$ and the Lagrangian multipliers is done in a single subroutine DCMINV with entry DXMINV. Subroutine DCMINV is an inversion routine for symmetric matrices. The special entry DXMINV uses the special block structure of the matrix for a fast solution. Method of solution is by elimination selecting the pivot on the diagonal each stage, with determination of the (effective) rank of the matrix. In case of a rank defect all remaining rows and columns of the resulting matrix V and the corresponding elements of B are set to zero.
As noted above, the separation of rows/columns related to measured and unmeasured variables is not done in the matrix inversion subroutine DCMINV, and the calculation of the elements $\boldsymbol{C}_{11} \ldots \boldsymbol{C}_{33}$ is not done according to the given formulae. Input to the matrix inversion routine (entry DXMINV) is the symmetric matrix

$$\begin{pmatrix} \tilde{\boldsymbol{V}}_x & \boldsymbol{A}^{\mathrm{T}} \\ \boldsymbol{A} & \boldsymbol{0} \end{pmatrix} ,$$

where the matrix $\tilde{\boldsymbol{V}}_x$ has zero entries for the unmeasured variables, and the derivative matrix corresponds to all (measured and unmeasured) variables. The matrix is stored in *symmetric storage mode*, where only the elements on and below the diagonal are stored. Result of the subroutine DCMINV is the inverse matrix (18), and the corrections $\Delta \boldsymbol{x}$. After convergence the inverse matrix (18) contains, in the submatrices $\boldsymbol{C}_{11}$, $\boldsymbol{C}_{12}$ and $\boldsymbol{C}_{22}$ the covariance matrix of the variables.

4.2 Numerical differentiation

The fast and accurate solution of the minimization problem requires the determination of the derivatives of the constraint equations w.r.t. all variables. The matrix of first derivatives, the Jacobian, has elements

$$\frac{\partial F_k}{\partial x_j} = (\boldsymbol{A})_{jk} \equiv \text{ A(J,K)} \; .$$

For linear constraint equations the derivatives are constants and a single step is sufficient for the solution. It is recommended to perform at least two steps. The second step allows to check the remaining values of the constraints; they should be at the round-off level. For non-linear constraint equations several steps are necessary, each step requires a redetermination of the derivative matrix $\boldsymbol{A}$. Good starting values of the parameters (unmeasured variables) allow usually a fast convergence, within few iterations, while bad starting values can lead to non-convergence.
The approximation of derivatives by *finite-difference* methods is a *local* method. It can be an ill-posed problem; the approximate values are affected by truncation errors and round-off errors. An alternative to the local methods are spectral or *pseudo-spectral* methods, that make use of a *global* representation. e.g. by a higher-order polynomial. Numerical methods for the estimation of derivatives are discussed in mathematical text books (e.g. [8]). In the following formulae, error estimates and hints for the optimal choice of step sizes for derivatives of a function $f(x)$ are given for local and semi-spectral methods.

Symmetric formula for the first derivative. The symmetric formula for the approximate determination of the first derivative requires two function values, $f(x+h)$ and $f(x-h)$ with a step size h. The function values, represented by floating point number, are affected by round-off errors, which can become relatively large, if the difference of the two function values is calculated. The relative error for the floating point representation is denoted by ε. For double-precision floating point numbers the numerical value for the uppper bound of ε is $6 \cdot 10^{-16}$; for practical calculations of an optimal step size a value of $7 \cdot 10^{-17}$ is usually more realistic. The step size h should be not too small to keep the round-off error small. The noise level of computed function values[30] limits the accuracy of numerically computed derivative values.
The symmetric formula for the estimate of the first derivative from the two function values $f(x+h)$ and $f(x-h)$ is

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h} \; . \tag{22}$$

A truncation error is introduced by non-zero higher derivatives of the function. These truncation errors can be estimated from the Taylor expansion of the function. The symmetric formula with two function values has the advantage that the second-derivative term does not influence the estimate (the estimate $f'(x) \approx (f(x+h) - f(x))/h$ with one function value is affected by the second derivative); the first term affecting the estimate is from the third derivative of the function.
An upper bound for the error of the derivative approximation is the sum of two terms from truncation and round-off:

$$\left| f'(x) - \frac{f(x+h) - f(x-h)}{2h} \right| \leq \frac{h^2}{6} M_1 + \frac{\varepsilon}{h} M_2 \tag{23}$$

with

$$M_1 = \max |f'''(\xi)|, \quad \xi \in [x-h, x+h] \qquad\qquad M_2 = \max |f(\xi)|, \quad \xi \in [x-h, x+h] \tag{24}$$

An optimal value for the step size h is obtained, if the two terms have the same magnitude:

$$h = \sqrt[3]{\frac{3\varepsilon M_2}{M_1}} \approx \sqrt[3]{\frac{3\varepsilon|f(x)|}{|f'''(x)|}} \tag{25}$$

where the bounds M_1 and M_2 are estimated by the values at x.

Pseudo-spectral Derivatives. An alternative to the local symmetric formulae from finite element methods above is based on the derivatives of a higher order polynomial, interpolating the function $f(x)$ at a set of nodes. Additional function values require additional Cpu-time, but can allow larger steps with a higher accuracy of the estimates of derivatives and should be less affected by truncation. Equally spaced nodes do not yield stable polynomial approximations, and therefore the case of Chebyshev polynomials with non-uniform Gauss-Lobato nodes is considered. For simplicity the interval $[-1, +1]$ is assumed, and the main interest are the derivatives in the center. The $(n+1)$ Gauss-Lobato nodes are (for $n \geq 1$)

$$z_j = -\cos\frac{\pi j}{n} \qquad 0 \leq j \leq n\ , \tag{26}$$

which correspond to the positions of the $(n+1)$ extrema (maxima and minima) of the n-th Chebyshev polynomial $T_n(z)$. The function values at the nodes have to be determined;

$$g_j = f(x + z_j h) \qquad 0 \leq j \leq n\ , \tag{27}$$

they form a vector $\boldsymbol{g}$ with $(n+1)$ elements. The calculation of the vector $\boldsymbol{g}'$ of derivatives g'_j is simple: it is obtained by the product $\boldsymbol{g}' = \boldsymbol{D}\boldsymbol{g}$ of a matrix $\boldsymbol{D}$ and the vector $\boldsymbol{g}$ (for details [8, 9]).
An interesting case is the value $n = 4$ which requires four function-value evaluations in addition to the central value $f(x)$. For the value $n = 4$ the five nodes are

$$z_0 = -1 \qquad z_1 = -1/\sqrt{2} \qquad z_2 = 0 \qquad z_3 = +1/\sqrt{2} \qquad z_4 = +1 \tag{28}$$

The formula for the first derivative in the center is

$$f'(x) \approx \frac{g_0 - 2\sqrt{2}\,(g_1 - g_3) - g_4}{2h}\ .$$

Round-off errors become of course large for the higher derivatives, as is already visible from the numerical factors. Nevertheless these approximate values of the higher derivatives can help in the determination of optimal step values, required for the finite-difference calculation of the first and second derivative.

4.3 Work storage

Work storage from argument WORK is subdivided into several subarrays. This subdivision is documented in the Table 2.

5 Use of analytical derivatives

The basic subroutine for numerical differentiation, step calculation and convergence control is subroutine APC, which is described in section 2. The fit can also be performed with anaytical derivatives, which may be more accurate and use less Cpu-time. This alternative requires two extra arrays.

```
DOUBLE PRECISION A(NX,NF)   ! derivative matrix (Jacobian)
DOUBLE PRECISION DX(NX)     ! step vector
```

nr of words	meaning
NX	saved initial variable array X(.)
NSYM=(N*N+N)/2	matrix W(.) with NSYM elements
N=NX+NF	right hand side of matrix equation
NF	corrected copy of constraints F(.)
N=NX+NF	corrections delta X
N=NX+NF	initial values of diagonal elements of matrix W
N=NX+NF	pointer to diagonal elements of matrix W
5*NF	constraint values during numerical differentiation
NF	reserved
(N*N+N)/2+5*N+5*NX	total number of words in array WORK(.)

Table 2: Subdivision of the work array WORK(.) into vectors and matrices.

Analytical derivative calculation. Below the code part for the calculation of analytical derivatives is shown. All elements of the Jacobian, matrix $\boldsymbol{A} \equiv$ A(.,.) have to be calculated. After the calculation APCALC is called, to calculate the step vector $\boldsymbol{\Delta x} \equiv$ DX(.), but not yet added to the variable vector $\boldsymbol{x} \equiv$ X(.); this is done by the following call of APCLIN, which also checks the convergence. The flag INI is not equal to zero, if convergence is not yet reached.

```
      IP=0    ! quiet       IP=1 for printout
      INI=0   ! reset iteration counter
10    F(1)    = ...
      ...
      F(NF)   = ...
      A(1,1)  = ...
      ...
      A(NX,NF)= ...

      CALL APCALC(INI,NX,X,VX, NF,F,A, DX,STATUS,IP, WORK)
      CALL APCLIN(INI,NX,X,DX,STATUS,  WORK)
      IF(INI.NE.0) GOTO 10
```

Instead of APCLIN, which is a version for almost linear problems, the user can call APCNON for non-linear problems.
The flag INI has to be reset to zero at the beginning, i.e. before the first step. Internally this flag is an initialization flag inside the package. The preparation for a minimization step includes the calculation of the constraints with the actual value of the variables X(.) and of the elements of the derivative matrix A().

The calculation of the step DX(.) is done in the entry APCALC. The user has to provide the arguments INI,NX,X,VX, NF,F,A and the work array WORK(.):

```
CALL APCALC(INI,NX,X,VX, NF,F,A, DX,STATUS,NDF, IP,WORK)
```

The flag INI is increased by 1 at each call, i.e. INI counts the iterations. Optionally one line with status information is printed for value IP=1, printout is suppressed for IP=0. The corrections DX(.) are determined, but not yet added to the variables X(.). Corrections are always defined with respect to the initial values of the variables X(.). Two different algorithms can be selected for the actual update of the variables by the corrections.

Variable update for simple (linear) fits. In case of linear constraints in principle a single step is sufficient to obtain the solution, but a second step allows to check the solution. In APCLIN the variables are updated and it is checked wether convergence is reached.

```
CALL APCLIN(INI,NX,X,DX,STATUS,NDF, WORK)
IF(INI.NE.0) GOTO 10
```

Convergence is indicated by the value zero of the flag INI. For the recognition of convergence the χ^2-values and the quantity $\sum_j |F_j|/NF$ (average absolute value of constrainsts) from the last two iterations are checked. At least two iterations are performed.

Variable update for non-linear fits. In the case of non-linear constraints the quality of initial approximations of the parameters is important for the speed of convergence. Divergence is possible for insufficient accuracy of the start values. The algorithm in APCNON is not optimized for speed, but it should be safer in difficult fits compared to APCLIN.

```
CALL APCNON(INI,NX,X,DX,STATUS,NDF, WORK)
IF(INI.NE.0) GOTO 10
```

If divergence is detected the algorithm will step back to previous values, with a reduced size of the steps.

6 Examples

Several examples are shown in this section. The source code of the examples is in the file apexamples.F. In all examples the macro apcom.inc shown on page 7 in section 2.1 for the declaration of the standard APCALC -variables is included.

6.1 Pythagorean theorem

A simple non-linear problem is solved in the subroutine APCHEK. The data x_1, x_2 and x_3 are measurements of the sides of a right angled triangle:

$$x_1 = 3.1 \pm 0.1$$
$$x_2 = 4.1 \pm 0.2$$
$$x_3 = 5.1 \pm 0.1$$

The relation $F_1(x) = x_1^2 + x_2^2 - x_3^2 = 0$, the Pythagorean theorem, is valid for a right angled triangle. The validity is forced by the code below

```
      SUBROUTINE APYTH      ! Pythagoras constraint
#include "apcom.inc"
      DOUBLE PRECISION DX(NXMAX),A(NXMAX,NFMAX)
      DOUBLE PRECISION Y(NXMAX),VY((NXMAX*NXMAX+NXMAX)/2),PULL(NXMAX)
      INTEGER I
*     ...
      WRITE(*,*) ' '
      WRITE(*,*) 'apyth: Pythagoras constraint'
      WRITE(*,*) '=============================='
      WRITE(*,*) 'Forcing validity of Pythagoras'
      NX=3
```

```
      NF=1
      CALL APCERO(NX,X,VX,NF,F)
      VX(1)= 0.1D0**2
      VX(3)= 0.2D0**2
      VX(6)= 0.1D0**2
      X(1)=3.1D0
      X(2)=4.1D0
      X(3)=5.1D0

 10   F(1)=X(1)*X(1)+X(2)*X(2)-X(3)*X(3)       ! <-- constraint
      CALL APC(NX,X,VX,NF,F,STATUS,  1,IST,WORK)
      IF(IST.NE.0) GOTO 10
      CALL APCRES(NX,X,VX, WORK)           ! print result

      CALL APCPUL(NX,VX,DX,PULL, WORK)     ! get pulls
      CALL APCOVA(NX,  VX, WORK)           ! get fitted VX(.)
      CALL APRXVX(NX,X,VX)                 ! print result
      CALL APCORR(NX,  VX)                 ! print correlations
      CALL TETIME('apyth ending')
      END
```

The output of the program is:

```
_____________________________________________
Constrained fit by apcalc with NX, NF=     3     1
_____________________________________________________________________________
 iter   ndf       chi^2_____delta          |F|/NF_________delta        ratio
   1      1       0.041                    0.4100
   2      1       0.041    0.2004E-03      0.1003E-02   -0.4090       -2.61
   3      1       0.041   -0.1512E-08      0.7519E-08   -0.1003E-02   -4.76
        21 F-evals (ISP=1, 3-point num.diff. calculation)

 par_i  fitted X_i_____+-std.dev.      initial X_i_____+-std.dev       pull
     1     3.09379       0.951857E-01       3.10000       0.100000     -0.20
     2     4.06734       0.118381           4.10000       0.200000     -0.20
     3     5.11026       0.862333E-01       5.10000       0.100000      0.20

 par_i        X_i_____+-std.dev.
     1     3.09379       0.951857E-01
     2     4.06734       0.118381
     3     5.11026       0.862333E-01

 Correlation coefficients between parameters i and j:
 par_i  \ j=     1     2     3
     1       1.000
     2      -0.439 1.000
     3       0.189 0.800 1.000
```

6.2 Error propagation

This examples demonstrates, that APCALC can be used for error propagation. Given are the rectangular coordinates $(9.0 \pm 0.1,\ 16.0 \pm 0.2)$, uncorrelated, of a point in a plane plane. The polar coordinates r, ϕ are needed. Part of a program is shown below

```
      NX=4
      NF=2
```

```
      CALL APCERO(NX,X,VX,NF,F)
      X(1)= 9.0D0
      X(2)=16.0D0
      VX(1)=0.1**2
      VX(3)=0.2**2
      X(3)=SQRT(X(1)**2+X(2)**2)
      X(4)=ATAN2(X(2),X(1))
10    F(1)=X(3)-SQRT(X(1)**2+X(2)**2)
      F(2)=X(4)-ATAN2(X(2),X(1))
      CALL APC(NX,X,VX,NF,F,STATUS,  5,IST,WORK)
      IF(IST.NE.0) GOTO 10 ! continue
      CALL APCRES(NX,X,VX, WORK)        ! print result
```

that performs the transformation to polar coordinates. The polar coordinate values are correlated with a correlation coefficient of $\rho = 0.540$, as shown by the printout of the program:

```
 Constrained fit by apcalc with NX, NF=     4     2

 iter  ndf       chi^2_____delta          |F|/NF_____________delta        ratio
   1     0       0.000                   0.8413E-16
   2     0       0.000     0.000         0.8413E-16     0.000          0.00
        34 F-evals (ISP=5, 5-point num.diff. calculation)

 par_i  fitted X_i_____+-std.dev.      initial X_i_____+-std.dev      pull
     1     9.00000      0.100000          9.00000      0.100000
     2     16.0000      0.200000          16.0000      0.200000
     3     18.3576      0.181078          18.3576
     4     1.05841      0.714635E-02      1.05841

 Correlation coefficients between parameters i and j:
 par_i  \ j=     1     2     3     4
     1       1.000
     2      -0.000 1.000
     3       0.271 0.963 1.000
     4      -0.664 0.747 0.540 1.000
 Transformed coordinates: radius and angle

 par_i         X_i_____+-std.dev.
     1     18.3576      0.181078
     2     1.05841      0.714635E-02

 Correlation coefficients between parameters i and j:
 par_i  \ j=     1     2
     1       1.000
     2       0.540 1.000
 Elements of covariance matrix:
  0.32789E-01 0.69829E-03 0.51070E-04
```

Of course, no fit is necessary in this case ($\chi^2 = 0$) and the rectangular coordinates remain unchanged.

6.3 Measurement of masses

The masses of two bodies are measured independently, they are x_1 and x_2; in addition the total mass x_3 of the two bodies is measured. A single constraint $F_1 = \hat{x}_1 + \hat{x}_2 - \hat{x}_3$ is applied in order to get

improved values of the two masses. The measured value (left) and the fitted values (right) are:

$$\begin{array}{rcl@{\qquad\Longrightarrow\qquad}rcl} x_1 & = & 101 \pm 1 & x_1 & = & 100.67 \pm 0.82 \\ x_2 & = & 99 \pm 1 & x_2 & = & 98.67 \pm 0.82 \\ x_1 + x_2 = x_3 & = & 199 \pm 1 & x_3 & = & 199.33 \pm 0.82 \end{array}$$

In the next example the difference x_4 of the two masses is measured (measurement x_4) too with higher precision. This allows a second constraint $F_2 = \widehat{x}_4 - (\widehat{x}_1 - \widehat{x}_2)$:

$$\begin{array}{rcl@{\qquad\Longrightarrow\qquad}rcll} x_1 & = & 101 \pm 1 & x_1 & = & 100.62 & \pm 0.41 \\ x_2 & = & 99 \pm 1 & x_2 & = & 98.72 & \pm 0.41 \\ x_1 + x_2 = x_3 & = & 199 \pm 1 & x_3 & = & 199.33 & \pm 0.82 \\ x_1 - x_2 = x_4 & = & 1.9 \pm 0.1 & x_4 & = & 1.9005 & \pm 0.0997 \end{array}$$

The result is a higher accuracy for the two mass values, while the measured difference remains almost unchanged.

6.4 Peelles problem

Data combination by constraints

A sequence of four data combination problems is discussed. A physical quantity is measured in two experiments, both measured values have statistical and additional systematic uncertainties, uncorrelated or correlated. The discussion shows, that the result can be very sensitive to rather small differencesof the statistical asssumptions, and shows the importance of a deep understanding of the measurement process.
The two measured data are x_1 and x_2. Averaging is obtained by the single constraint

$$F_1 = x_1 - x_2$$

which means that the *fitted* values of the two variables have to become equal with $F_1 = 0$.

Two values with constant (Gaussian) absolute uncertainty

The first problem is rather simple. It will show the application of the least squares method with constraints for data combination.

> **Problem 1** *A physical quantity is measured independently in two experiments with the data:*
>
> $$\begin{aligned} &\mathit{Measurement}\ 1 : m_1 = 1.5 \pm 0.15 \\ &\mathit{Measurement}\ 2 : m_1 = 1.0 \pm 0.10 \end{aligned}$$

The description of the measurement result is interpreted to mean two values with *constant* (Gaussian) variance. The simple (unweighted) average $(m_1 + m_2)/2$ is 1.25. The weighted average will give a slightly smaller result, because the lower of the two values has a higher accuracy and thus a larger weight. The covariance matrix for the two measured values is diagonal:

$$\boldsymbol{V} = \begin{pmatrix} 0.15^2 & 0 \\ 0 & 0.10^2 \end{pmatrix}$$

and the problem is a *linear* least squares problem, which is treated in the following small program:

```
      NX=2
      NF=1
      CALL APCERO(NX,X,VX,NF,F)      ! reset
      X(1)=1.5
      VX(1)=0.15**2
      X(2)=1.0
      VX(3)=0.10**2

20    F(1)=X(1)-X(2)
      CALL APC(NX,X,VX,NF,F,STATUS,1,IST, WORK)
      IF(IST.NE.0) GOTO 20           ! continue

      CALL APCRES(NX,X,VX, WORK)     ! print
```

The printout shows the iterations and the result.

```
iter   ndf       chi^2_____delta          |F|/NF_________delta       ratio
  1      1       7.692                    0.5000
  2      1       7.692     0.000           0.000       -0.5000      -7.70
       10 F-evals (ISP=1, 3-point num.diff. calculation)

par_i  fitted X_i_____+-std.dev.      initial X_i_____+-std.dev      pull
    1     1.15385      0.832050E-01       1.50000      0.150000     -2.77
    2     1.15385      0.832050E-01       1.00000      0.100000      2.77
```

The result is, as expected, slightly lower than 1.25, the simple average[3]:

$$\overline{m} = 1.15385 \pm 0.08320$$

The linear fit requires in principle only one step. The final result is in fact already reached after one step, but two iterations are used, to verify convergence. After the fit the two values agree, and have pulls of the same size, but opposite sign. The square of the pull of 2.77 is equal to the χ^2-value of the fit of 7.7, which corresponds to a rather small p-value of 0.5%.

Two values with constant relative uncertainty

The next problem is a small modification of the previous problem. Here *relative* uncertainties of 10% are given for both values (for the actual measured values the uncertainties are identical to the previous problem).

> **Problem 2** *A physical quantity is measured independently in two experiments with the data:*
>
> $$\mathit{Measurement}\,1 : m_1 = 1.5 \pm 10\%$$
> $$\mathit{Measurement}\,2 : m_1 = 1.0 \pm 10\%$$

The fact that the uncertainties are given as relative, means that the uncertainties are not constant and the conditions for a linear least squares fit are not given. Instead a log-normal distribution is assumed, which means that the logarithms of the values have a constant variance. A third variable x_3 as the average value is introduced; this is only a formal change, and requires two constraints (instead

[3]Here and in other problems the result is given with more digits than usual for the result in order to allow the comparison of different solutions for the same problem.

of one). The relative uncertainty ε_{rel} for a variable can be described by a factor $\exp(0 \pm \varepsilon_{\text{rel}})$, i.e. the measured value $1.5 \pm 10\%$ is expressed by $1.5\exp(x_1)$ with $x_1 = 0 \pm 0.10$. This treatment is used in the code

```
      NX=3
      NF=2
      CALL APCERO(NX,X,VX,NF,F)      ! reset
      X(1)=0.0
      VX(1)=0.1**2
      X(2)=0.0
      VX(3)=0.10**2
      X(3)=1.0
20    F(1)=1.5*EXP(X(1))-X(3)
      F(2)=1.0*EXP(X(2))-X(3)
      CALL APC(NX,X,VX,NF,F,STATUS,1,IST, WORK)
      IF(IST.NE.0) GOTO 20           ! continue
      CALL APCRES(NX,X,VX, WORK)     ! print
```

with the printout:

```
iter  ndf       chi^2_____delta          |F|/NF_________delta       ratio
  1     1       7.692                    0.2500
  2     1       8.252     0.5594         0.2475E-01   -0.2252      -1.00
  3     1       8.220    -0.3168E-01     0.8543E-03   -0.2390E-01  -1.46
  4     1       8.220     0.6616E-04     0.3032E-05   -0.8513E-03  -2.45
  5     1       8.220     0.5173E-08     0.3622E-08   -0.3028E-05  -2.35
       35 F-evals (ISP=1, 3-point num.diff. calculation)

par_i  fitted X_i_____+-std.dev.      initial X_i_____+-std.dev      pull
    1   -0.202733      0.707107E-01       0.00000      0.100000    -2.87
    2    0.202732      0.707107E-01       0.00000      0.100000     2.87
    3     1.22474      0.866025E-01       1.00000
```

The result is now

$$\overline{m} = 1.22474 \pm 0.08660$$

and rather close to 1.25, the simple average. The relative uncertainty $0.08660/1.22474 = 0.07071$, calculated in APCALC , corresponds to $10\%/\sqrt{2}$, as expected for the case of two identical (relative) uncertainties. The difference between the previous result and this result is about one standard deviation and thus not small. An accurate knowledge of the experimental method is required to choose one of the alternative assumptions. The latter assumptions seems to be appropriate, because relative uncertainties are given. Because of the nonlinearity the problems requires more than two iterations. The constraints are approaching zero rather fast, and the χ^2 changes become rather small.

Two values with common normalization uncertainty

The two problems treated above are an introduction to the next problem, which has been discussed in 1987 by Robert Peelle (Oak Ridge National Laboratory) and has triggered several publications, even recently, especially in the nuclear data community, because of the strange result with an average outside the range of the two data. The problem is given in the *original* text of R. Peelle[18].

> **Problem 3** *"Suppose we are required to obtain the weighted average of two experimental results for the same physical quantity. The first result is 1.5, and the second result 1.0. The full covariance matrix of those data is believed to be the sum of three*

components. The first component is fully correlated with standard error 20% of each respective value. The second and third components are independent of the first and of each other, and correspond to 10% random uncertainties in each experimental result."

Apparently two values 1.5 and 1.0 are measured independently with a relative uncertainty of 10%. In addition there is a *common* uncertainty of 20%, which can be interpreted as a common normalization uncertainty for the two measured values. Without any calculation one can conclude, that the average value should be identical to the result of problem 2 before, because the process of averaging in this case does not provide any information on the common normalization. Only the uncertainty will become larger, if the normalization uncertainty is included.

This problem is instructive, because its solution is discussed in many publications. The original text above already gives an indication on the – wrong – solution method, used for this problem with two correlated date: it is solved as a **linear** least squares problem, with a definition of a non-diagonal covariance matrix, although the problem is a **non-linear** problem. The calculated result is given and discussed in the paper [18]:

'The weighted average obtained from the least-squares method is 0.88 ± 0.22*, a value outside the range of the input values! Under what conditions is this the reasonable result that we sought to achieve by use of an advanced data reduction technique?"*

In the paper the average is determined with the minimization of the χ^2-expression. The covariance matrix $\boldsymbol{V}$ is constructed to include the normalization uncertainty.

$$\begin{aligned}\boldsymbol{V} &= \begin{pmatrix} \sigma_1^2 + \epsilon^2 x_1^2 & \epsilon^2 x_1 x_2 \\ \epsilon^2 x_1 x_2 & \sigma_2^2 + \epsilon^2 x_2^2 \end{pmatrix} \\ &= \begin{pmatrix} 0.15^2 & 0 \\ 0 & 0.10^2 \end{pmatrix} + \begin{pmatrix} 0.30^2 & 0.30 \times 0.20 \\ 0.30 \times 0.20 & 0.20^2 \end{pmatrix} = \begin{pmatrix} 0.1125 & 0.06 \\ 0.06 & 0.05 \end{pmatrix}\end{aligned}$$

The somewhat different measured values for x_1 and x_2 are used to calculate the contribution of the normalization factor, and this causes a deviation from symmetry into the matrix with different diagonal elements (main axis of the covariance ellipse tilted w.r.t. the diagonal). As mentioned, the problem was reported in 1987 as a case of an *anomalous* result from standard least squares. The apparently *strange* least squares result in the original publication[18] is discussed in conferences and a large number of publications, especially in the nuclear physics community.
The treatment of the Peelle problem with APCALC shows, that acceptable results can be obtained without any difficulty by constrained least squares, The code includes now the normalization factor $x_4 = 1.0 \pm 0.20$, assumed to be log-normal distributed. The nonlinearity is included in the equality constraint. Using APCALC the solution is straightforward and simple.

```
NX=4
NF=2
CALL APCERO(NX,X,VX,NF,F)      ! reset
X(1)=0.0
VX(1)=0.1**2
X(2)=0.0
VX(3)=0.10**2
X(3)=1.0
```

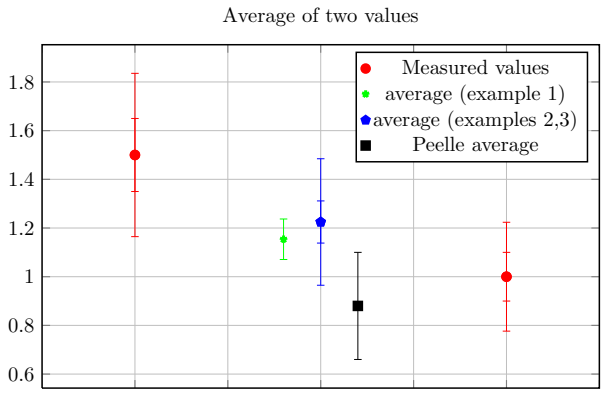

Figure 2: The measured values, the average values from examples 1, 2 and 3, and the Peelle average.

```
      X(4)=0.0
      VX((4*4+4)/2)=0.20**2
20    F(1)=1.5*EXP(X(1))*EXP(X(4))-X(3)
      F(2)=1.0*EXP(X(2))*EXP(X(4))-X(3)
      CALL APC(NX,X,VX,NF,F,STATUS,1,IST, WORK)
      IF(IST.NE.0) GOTO 20            ! continue
      CALL APCRES(NX,X,VX, WORK)      ! print
```

with output

par_i	fitted X_i_____+-std.dev.	initial X_i_____+-std.dev	pull
1	−0.202733 0.707107E−01	0.00000 0.100000	−2.87
2	0.202733 0.707107E−01	0.00000 0.100000	2.87
3	1.22474 0.259808	1.00000	
4	0.593405E−07 0.200000	0.00000 0.200000	0.00

The result is as expected: the average value is identical to the previous value, and the uncertainty calculated by APCALC is larger:

$$\overline{m} = 1.22474 \pm 0.25981$$

All averages are shown in Figure 2. The error bars are the statistical (uncorrelated uncertainty) and the total (quadratic addition of uncorrelated and correlated) uncertainties. No contribution of the correlated uncertainty is shown for the average from example 1. The error bar for the Peelle avarage is the total uncertainty; this value is smaller than the uncertainty from example 3. It is easy to see that the uncertainty of the APCALC average is equal to the previous uncertainty and the 20% normalization uncertainty, added quadratically: $\pm 0.25981 = \pm 0.08660 \pm 20\%$.

The printer output from APCALC gives further information about the problem and its solution. The normalization factor x_4 is not modified by the fit, as expected, and the pull of x_4 is zero, as expected, because the normalization factor does not contribute to the process of averaging. The measured values x_1 and x_2 *after* the fit are identical to the case of problem 2.

Different methods are discussed in reference [16], and the results are shown in Table 3. The first five rows, obtained by different χ^2-sum methods are from Reference[16], the last row gives the result from constrained least squares with APCALC . Methods (B) to (D) treat the correlation effect as multiplicative, in different ways. Often a solution is searched for in Bayesian Statistic. A recently published detailed discussion[17] says: "...this phenomenon was attributed to the underlying non-

Average	uncertainty	comment
0.882	± 0.228	(A) PPP - additive
1.200	± 0.276	(B) normalization error
1.177	± 0.253	(C) normalization error, log-normal prior
1.252	± 0.267	(D) logarithmic analysis
1.041	± 0.295	(A + B)
1.225	± 0.260	APCALC result

Table 3: Results from the analysis of Peelles problem[18], the combination of the two values $1.5 \pm 10\%$ and $1.0 \pm 10\%$, with a common normalization uncertainy of 20%. See text for further explanation.

linearity of the relation between data. Here, we show in terms of Bayesian Statistics that Peelle's Pertinent Puzzle is primarily caused by improper estimates of covariance matrices of experiments and not exclusively by non-linearities."[17]

It may be helpful in cases of systematic uncertainties in addition to statistical uncertainties to consider the Monte Carlo strategy necessary to simulate the measurement process realistically with all uncertainties in detail, and perhaps to perform the Monte Carlo simulation. In reference[15] the detailed data analysis is called "a dialogue with the data", with the statement: "All analyses are conditional on assumptions and approximations, and it's important to understand and state them clearly."

A variation of the Peelle pertinent puzzle

The same – incorrect – least squares treatment is discussed again 20 years later in a paper by G.d'Agostini[19]. The problem is almost identical to the Peelle problem.

> **Problem 4** *"... the result of two measurements,* $8.0 \pm 2\%$ *and* $8.5 \pm 2\%$*, have a 10 % commmon normalization error. Assuming that the two measurements refer to the same physical quantity, the best estimate of its true value can be obtained by fitting the points to a constant function. Minimizing* χ^2 *... with* $\boldsymbol{V}$ *estimated empirically from the data ..." [19]*

In the paper the linear least square method is applied with constant (Gaussian) covariance matrix, the result is 7.87 ± 0.81, again outside the range of the two values. From the paper[19]: The abstract says: "Best fits to data which are affected by systematic uncertainties on the normalization factor have the tendency to produces curves lower than expected of the covariance matrix of the data points is used in the definition of the χ^2. This paper shows that the effect is a direct consequence of the hypothesis used to estimate the empirical covariance matrix, namely the linearization on which the ususal error propagation relies." "It has also been shown that this bias comes from the linearization performed in the usual covariance propagation. This means that, even though the use of the covariance matrix can be very useful in analysing the data in a compact way using available computer algorithms, care is required if there is one large normalization uncertainty which affects all the data".

The result obtained by APCALC with a code almost identical to the code from the previous problem is:

$$8.24621 \pm 0.11662 \pm 10\% = 8.24621 \pm 0.83283 \ ,$$

well inside the range of the two measured values.

6.5 Combination of correlated measurements

Gaussian data

Methods for the combination of correlated measurements of different physical quantities are discussed by Valassi[10], using a fictitious example. The technique reviewed in the paper requires a positive definite covariane matrix, which does not depend on the result of the measurement, i.e. the measured values have to be multivariate Gaussian distributed. Under these conditions the least square formalism delivers the "best linear unbiased estimate" (BLUE). All problems given in the paper can be solved by APCALC with two or three simple linear constraints.
The case of two experiments A and B, measuring the branching fraction of the W boson in the two decay channels to electrons and tau leptons is assumed. The four measurements are:

Branching fraction to electron	(A)	$x_1 = 0.105 \pm 0.010$
Branching fraction to electron	(B)	$x_2 = 0.135 \pm 0.030$
Branching fraction to tau	(A)	$x_3 = 0.095 \pm 0.030$
Branching fraction to tau	(B)	$x_4 = 0.140 \pm 0.030$

The data from experiments A and B can be combined, separately for the electron and the tau channel, and, assuming lepton universality, for both channels.

No lepton universality assumed. The two branching ratios are treated as measurements of independent quantities. Two constraints are used which force the equality of the two electron measurements and the two tau measurements:

$$F_1 = x_1 - x_2 \qquad F_2 = x_3 - x_4 \; .$$

```
 ------------------------------------------------------
 Constrained fit by apcalc with NX, NF=     4     2
 ------------------------------------------------------------------------------
  iter   ndf       chi^2_____delta          |F|/NF_________delta        ratio
    1      2       2.025                    0.3750E-01
    2      2       2.025    0.4441E-15       0.000       -0.3750E-01   -6.57
         18 F-evals (ISP=1, 3-point num. diff. calculation)

  par_i  fitted X_i_____+-std.dev.      initial X_i_____+-std.dev       pull
      1    0.108000      0.948683E-02     0.105000      0.100000E-01   0.95
      2    0.108000      0.948683E-02     0.135000      0.300000E-01  -0.95
      3    0.117500      0.212132E-01     0.950000E-01  0.300000E-01   1.06
      4    0.117500      0.212132E-01     0.140000      0.300000E-01  -1.06
```

The result of the fit is

$$x_1 = x_2 = 0.1080 \pm 0.0095$$
$$x_3 = x_4 = 0.1175 \pm 0.0212$$

Lepton universality assumed. In case of lepton universality the two branching ratios are identical. A third constraint forces the equality of the two branching ratios:

$$F_1 = x_1 - x_2 \qquad F_2 = x_3 - x_4 \qquad F_3 = x_1 - x_3$$

```
 Constrained fit by apcalc with NX, NF=      4      3
 ------------------------------------------------------------------------------
  iter   ndf       chi^2_____delta          |F|/NF_________delta        ratio
    1      3       2.192                    0.2833E-01
    2      3       2.192      0.000          0.000       -0.2833E-01    -6.45
         18 F-evals (ISP=1, 3-point num.diff. calculation)

  par_i  fitted X_i_____+-std.dev.      initial X_i_____+-std.dev       pull
      1    0.109583      0.866025E-02     0.105000      0.100000E-01   0.92
      2    0.109583      0.866025E-02     0.135000      0.300000E-01  -0.88
      3    0.109583      0.866025E-02     0.950000E-01  0.300000E-01   0.51
      4    0.109583      0.866025E-02     0.140000      0.300000E-01  -1.06
```

The result of the fit is

$$x_1 = x_2 = x_3 = x_4 = 0.1096 \pm 0.0087$$

Code. The program code for the data combination is given below. The essential code for the data combination are, in this trivial problem, the simple expressions for the definition of the constraints $F_1 ... F_3$. After the macro apcom.inc for the standard APCALC variables two data statement contain the measured values and the elements of the covariance matrix. The two fits (ICASE=1 and =2) differ only in the third constraint in ICASE=2.
The subroutine APBLUE(IARG) is among the example subroutines. The argument IARG can be used to choose different correlations by IARG=0 to IARG=3.

```
#include "apcom.inc"
      INTEGER I,ICASE
      DOUBLE PRECISION XDAT(4),COVM(10)
      DATA XDAT/0.1050D0,0.1350D0,0.0950,0.1400/
      DATA COVM/1.0D-4,0.0D-4,9.0D-4,
     *          0.0D-4,0.0D-4,9.0D-4,0.0D-4,0.0D-4,0.0D-4,9.0D-4/
      DO ICASE=1,2
       NX=4            ! number of variables
       NF=1+ICASE      ! ... constraints (without/with lepton universality
       DO I=1,4
        X(I)=XDAT(I)   ! copy data
       END DO
       DO I=1,10
        VX(I)=COVM(I) ! copy covariance matrix
       END DO
       ISP=1           ! printout 3-point_derivatives linear_problem
 10    CONTINUE
 20    F(1)=X(1)-X(2)
       F(2)=X(3)-X(4)
       IF(ICASE.EQ.2) F(3)=X(1)-X(3)    ! additional constraint
       CALL APC(NX,X,VX,NF,F,STATUS,ISP,IST,WORK)
       IF(IST.NE.0) GOTO 20
       CALL APCRES(NX,X,VX,WORK)
      END DO ! ICASE
      END
```

Many different cases with different correlations (non-zero elements of the off-diagonal matrix elements) are discussed in the paper[10]. All cases can be calculated ("best linear unbiased estimates") by choosing argument IARG or with a different data statement for COVM by the same code, with result identical to the published[10] results.

Non-Gaussian data

As pointed out by Valassi[10] "estimating the input covariance matrix is one of the most delicate steps in the combination of results." A detailed understanding of the physics and the measurement process is necessary to understand the statistical character of the statistical and systematic uncertainties.
An example for a "breakdown of error contributions" is given in the paper[10]. Assuming for experiment B very small statistical errors but large systematic errors, the measurements

Branching fraction to electron	(A)	$x_1 = 0.105 \pm 0.010$ stat
Branching fraction to electron	(B)	$x_2 = 0.1350 \pm 0.0021$ stat ± 0.0299 syst
Branching fraction to tau	(A)	$x_3 = 0.095 \pm 0.030$ stat
Branching fraction to tau	(B)	$x_4 = 0.1400 \pm 0.0021$ stat ± 0.0299 syst

are assumed. The covariance matrix is written as a sum of two contributions from statistical and systematic errors:

$$\boldsymbol{V} = \begin{pmatrix} 1.00 & 0 & 0 & 0 \\ 0 & 0.04 & 0 & 0 \\ 0 & 0 & 9.00 & 0 \\ 0 & 0 & 0 & 0.04 \end{pmatrix}_{\text{(stat)}} \times 10^{-4} + \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 8.96 & 0 & \pm 8.96 \\ 0 & 0 & 0 & 0 \\ 0 & \pm 8.96 & 0 & 8.96 \end{pmatrix}_{\text{(syst)}} \times 10^{-4}$$

The correlation between the two values from experiment B is either 100 % positive or 100 % negaive correlated. In the paper (constant) Gaussian measurement errors are assumed; the breakdown is used in the paper to split the errors of the fitted branching ratios into statistical and systematic contributions.
However the assumptions made above suggest that the statistical and the systematic uncertainties are non-Gaussian. The uncertainties are due to an event count, following a Poisson distribution, and a normalization factor, following a log-normal distribution. The number of events behind the value 0.1350 ± 0.0021 can be estimated as $(0.1350/0.0021)^2 = 4133$. This number is large enough to justify a Gaussian approximation. The relative normalization uncertainty can be estimated from the relative uncertainty $0.0299/0.1350 = 0.2215$, which suggests a log normal distributed uncertainty factor with relative uncertainty 0.22.
An extension to data combination under non-Gaussian conditions with nonlinearities is simple for APCALC . A fifth measured value $x_5 = 0 \pm 0.22$ is introduced and the constraint equations are changed to

$$F_1 = x_1 - x_2 \times \exp(x_5) \qquad F_2 = x_3 - x_4 \times \exp(\pm x_5) \; .$$

(the $\pm$ corresponds to 100 % positive/negative correlation). For a positive correlation the results from the Gaussian assumption [10] and the log-normal assumption with APCALC are

	Reference[10]	APCALC
electron	0.1064 ± 0.0091	0.1075 ± 0.0088
tau	0.1114 ± 0.0094	0.1115 ± 0.0093
lepton	0.1071 ± 0.0090	0.1081 ± 0.0088

and for the negative correlation

	Reference[10]	APCALC
electron	0.1144 ± 0.0091	0.1173 ± 0.0091
tau	0.1598 ± 0.0094	0.1601 ± 0.0120
lepton	0.1367 ± 0.0015	0.1367 ± 0.0015

The largest deviation between the two sets of values is 0.3 standard deviations. In cases with sevaral sources of systematic uncertainties the log-normal effects can be larger.

6.6 Histogram fit of a Gaussian peak

A Gaussian peak plus a constant background is fitted to a 100-bin histogram (Figure 3). The variables and the histogram parameters are:

normalizing factor	x_1
mean value	x_2
inverse variance	$x_3 \equiv \sigma^{-2}$
background	x_4
histogram bin contents	$x_5 \ldots x_{104}$
binwidth	Δ
bin center	$\tilde{x}_i,\ i = 1,\ 2 \ldots 100$

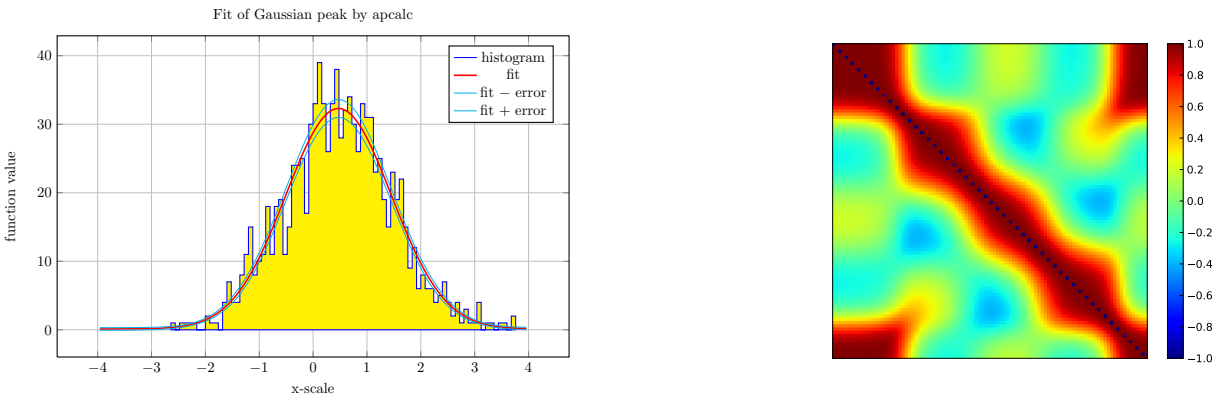

Figure 3: Fit of a histogram with a Gaussian peak. The fitted bin-entries with standard deviations, shown as a band around the fitted curve, are already calculated during the fit. The plot on the right shows the correlation coefficients of the 100×100 covariance matrix of the bin-entries.

The 100 constraint equations are the difference between the observed count x_{4+i} and the parametrization of the Gaussian (three parameters x_1, x_2, x_3) plus a constant background (x_4). Instead of the standard deviation σ of the Gaussian the inverse variance σ^{-2} is used in fit.

$$F_{4+i} = x_{4+i} - \left\{ \frac{x_1 x_3 \Delta}{\sqrt{2\pi}} \exp\left(-\frac{1}{2} x_3 \left(\tilde{x}_i - x_2 \right)^2 \right) + x_4 \right\} \qquad i = 1,\ 2 \ldots 100$$

Part of the program code is shown below. In the first part following the principle of IRLS, the variance of the bin contents is redefined at the start of each iteration (but the variance is not a fit parameter!). Thus the least squares result is equivalent to a maximum likelihood fit, based on the Poisson distribution for the bin contents.

```
      ITER=0
10    ITER=ITER+1                    ! next iteration
      DO I=1,100
       J=4+I
       VX((J*J+J)/2)=MAX(1.0,X(J))   ! POISSON variance (IRLS)
      END DO

20    DO I=1,100
       XX=XA+BINWID*(DFLOAT(I)-0.5D0)
       DERF=0.39894228D0*EXP(-0.5D0*(X(3)*(XX-X(2)))**2)
       F(I)=X(4+I)-(X(1)*X(3)*DERF*BINWID+X(4))
      END DO
      CALL APC(NX,X,VX,NF,F,STATUS,1,IST, WORK)
      IF(IST.GT.0) GOTO 10           ! continue iteration
      IF(IST.LT.0) GOTO 20           ! continue derivative calculation
```

In the second part the constraints are evaluated for the given values of the variables. Internally the derivative matrix is calculated, followed by the step calculation. The result of the fit is shown in Figure 3. The fitted curve with the $\pm 1\sigma$ band is already calculated in the fit; on the right the correlation coefficients ρ (with $-1 \leq \rho \leq +1$) are shown for the 100 bins.
In this example the bin content is, for simplicity, evaluated at the bin center $\tilde{x}_i$; this causes a bias in the fit result for narrow peaks. For serious applications this method has to be replaced by an accurate calculation of the integral over the bin. The result will be biased too in methods, based on the minimization of a χ^2-expression with a variance in the denominator given either by the measured bin content (Neyman χ^2) or the expected bin content (Pearson χ^2).

6.7 Straight line fit for $x-y$-data with uncertainties in both coordinates

The solution of complex problems with the standard least squares method using a χ^2-expression can become rather complicated. In this section the case of straight-line fits of (x, y) data with uncertainties *both* in x and in y is treated. Methods for least-squares fitting with such data are reviewed in detail in a Technical note[11]; a list of 32 publications on different proposed methods is provided in this note, which in general give different fit results. The example used in this section is taken from this note; the data from Pearson[13] with weights suggested by York[14], are called "well known", and are shown in Table 4. The weight given in the table is related to the standard deviation by $w_i = 1/\sigma_i^2$.

$x-y$-data with uncorrelated uncertainties in both coordinates. The subject is discussed by Press et. al. [12] (Numerical Recipes) with the remarks: *"If experimental data are subject to measurement error not only in the y_i's, but also in the x_i's, then the task of fitting a straight-line model $y(x) = a + bx$ is considerably harder. ... Be aware that the literature on the seemingly straightforward subject of this section is generally confusing and sometimes plain wrong."* Press et al. include a subroutine fitexy for the straight-line with uncorrelated $x-y$-data.
The solution is trivial, if constrained fitting is applied. For 10 data points 10 constraints are defined, by

$$F_i = a + b \cdot x_i - y_i \qquad i = 1,\, 2 \ldots 10\ . \tag{29}$$

with intercept a and slope b for the data of Table 4. The data x_i and y_i are stored in elements X(2*I−1) and X(2*I), and the variances are stored in corresponding elements of matrix VX(.). Intercept and slope are assigned to the elements X(21) and X(22). The part of the code for the fit is simply

	x	w_x	y	w_y
1	0.0	1000.0	5.9	1.0
2	0.9	1000.0	5.4	1.8
3	1.8	500.0	4.4	4.0
4	2.6	800.0	4.6	8.0
5	3.3	200.0	3.5	20.0
6	4.4	80.0	3.7	20.0
7	5.2	60.0	2.8	70.0
8	6.1	20.0	2.8	70.0
9	6.5	1.8	2.4	100.0
10	7.4	1.0	1.5	500.0

Table 4: Example data "Pearson's data with York's weights" for comparison of fitting procedures, from Cantrell[11]. Weights w are inverse variances.

```
20    DO I=1,10
       F(I)=X(21)+X(22)*X(2*I-1)-X(2*I)
      END DO
      CALL APC(NX,X,VX,NF,F,STATUS,1,IST, WORK)
      IF(IST.NE.0) GOTO 20           ! continue
```

This straight-line fit is a non-linear problem. With start values $a = 0$ and $b = 0$ for intercept and slope some iterations are necessary:

iter	ndf	chi^2	delta	|F|/NF	delta	ratio
1	8	34.345		3.700		
2	8	10.646	−23.70	0.9626E−10	−3.700	−8.56
3	8	11.872	1.226	0.1712E−01	0.1712E−01	6.23
4	8	11.866	−0.5614E−02	0.3625E−03	−0.1675E−01	−1.67
5	8	11.866	−0.4153E−04	0.2890E−06	−0.3622E−03	−3.08
6	8	11.866	−0.1973E−06	0.5723E−07	−0.2318E−06	−0.65
7	8	11.866	0.4021E−08	0.5420E−10	−0.5718E−07	−0.83

The result of the fit is shown below, together with the published result.

	this calculation			C.A. Cantrell [11]	
	value	st.dev.	st.err.	value	st.err.
slope	−0.4805302	±0.05799	±0.07062	−0.4805334	0.07062
intercept	5.4798943	±0.29497	±0.35925	5.4799102	0.35925
χ^2	11.8663532			11.8663532	

Note that in the published note, instead of standard deviation σ, a so called *standard error* $(= 1.218\sigma)$ is used. The factor 1.218 is equal to $\sqrt{\chi^2/(n-2)}$ for $n = 10$. Figure 4 shows the data and the fit. The pull values for the 10 data points are given below (the values for x and y are identical):

index	1	2	3	4	5	6	7	8	9	10
pull	−0.44	−0.50	+0.47	−1.16	+2.06	−1.57	+1.70	−1.96	−0.12	+0.98

The 5.th point has the largest deviation from the straight line, with a pull of +2.06.

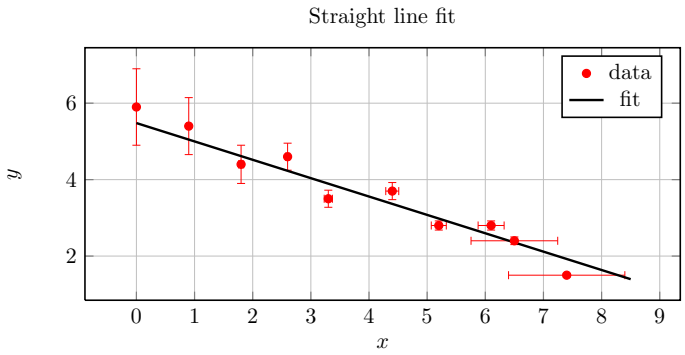

Figure 4: Straight-line fit to data points (x, y) with uncertainties in x and y.

$x-y$-data with correlated uncertainties in both coordinates. For APCALC the case where both x_i- and y_i-values have uncertainties and are *correlated* is still simple. Data points $P_j = (x_j, y_j)$ are considered, each with a 2×2 covariance matrix V_j, for $j = 1, 2 \ldots n$. The task is to determine the parameters a and b of the straight-line parametrization $y = a + b \cdot x$, with a minimum of the distance of each point $P_j = (x_j, y_j)$ to the straight-line, taking into account the covariance matrix V_j for each data point. The single-point data

$$P_j = \begin{pmatrix} x_j \\ y_j \end{pmatrix} \qquad V_j = \begin{pmatrix} \sigma_{x_j}^2 & \rho_{xy_j}\sigma_{x_j}\sigma_{y_j} \\ \rho_{xy_j}\sigma_{x_j}\sigma_{y_j} & \sigma_{y_j}^2 \end{pmatrix} \qquad j = 1, 2 \ldots n \tag{30}$$

are combined to a covariance matrix with $2n + 2$ rows and columns: 2 for each data point, and 2 for the parameters a and b with zero values of the corresponding elements of the covariance matrix.

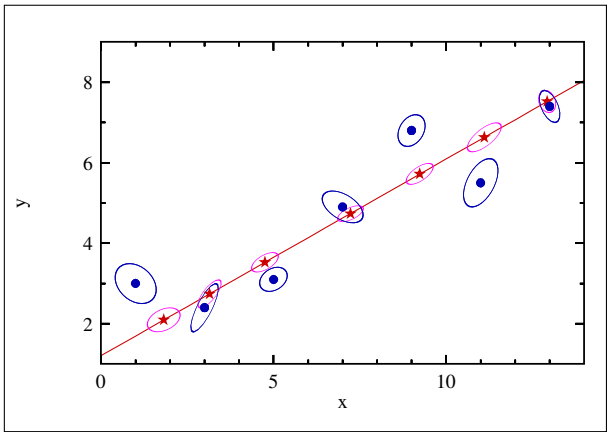

Figure 5: Straight-line fit to data points (x, y) with correlated errors, shown by the 1-σ ellipses, before and after the fit. The line is the fit, and the stars indicate the fitted point coordinates.

The constraint equations are *identical* to the case above (equation (29)) and the program code is identical too. Data points from an example and the fit results are plotted in Figure 5. The general-

ization to a fit of a parabola or other parametrizations requires only a corresponding change in the constraint equations.

6.8 ... last but not least

The example taken from [20] is in fact rather special. Two measurements with two correlated systematic uncertainties are given, but no individual uncertainty is assigned to the measurements.

> A quantity m is measured in two experiments. For both measurements only two *systematic* uncertainties are given.
>
> $$\text{Measurement } 1 : x_1 = 5 \pm 1\ (\text{syst1}) \pm 1\ (\text{syst2})$$
> $$\text{Measurement } 2 : x_2 = 5 \pm 1\ (\text{syst1}) \pm 2\ (\text{syst2})$$
>
> The systematic uncertainty 1 is 100% correlated and the systematic uncertainty 2 is 100% correlated[20] too.

The information on systematic uncertainties is rather strange. In the calculation below both uncertainties are assumed to be additive. The systematic uncertainty 1 is included in the covariance matrix and the systematic uncertainty 2 is described by an extra measured value x_3, additive to x_1 with factor 1 and to x_2 with factor 2. The code and the result listing are shown below.

```
      NX=4
      NF=2
      CALL APCERO(NX,X,VX,NF,F)      ! reset
      X(1)=5.0
      X(2)=5.0
      VX(1)=1.0
      VX(2)=1.0                      ! 100 % correlated
      VX(3)=1.0
      X(3)=0.0
      VX(6)=1.0
      X(4)=4.0                       ! initial approx.
20    F(1)=X(1)+      X(3) -X(4)
      F(2)=X(2)+2.0D0*X(3) -X(4)
      CALL APC(NX,X,VX,NF,F,STATUS,1,IST, WORK)
      IF(IST.NE.0) GOTO 20           ! continue
      CALL APCRES(NX,X,VX, WORK)     ! print
```

```
iter  ndf       chi^2_____delta          |F|/NF_________delta       ratio
  1     1       0.000                     1.000
  2     1       0.000     0.000           0.000        -1.000       -8.00
       18 F-evals (ISP=1, 3-point num.diff. calculation)

par_i  fitted X_i_____+-std.dev.      initial X_i_____+-std.dev      pull
    1     5.00000       1.00000          5.00000       1.00000
    2     5.00000       1.00000          5.00000       1.00000
    3     0.00000      0.278775E-07      0.00000       1.00000
    4     5.00000       1.00000          4.00000
```

The result is

$$\text{average } = 5.0 \pm 1.0$$

which is also quoted in [20], where the average is called *agressive*, perhaps because of the apparent reduction in uncertainty by the averaging fit. A closer look at the special uncertainty structure explains the result. Because the measured values agree accurately, only the systematic uncertainty component with the same factor for both measurements is active, but the other uncertainty is excluded (zero). Note that the input covariance matrix is singular and thus no inverse exists; the χ^2 and all pulls are zero. Nevertheless APCALC is able to solve this exotic problem.
The situation changes, if the two measured values do not agree. If the two measured values are 4.5 and 5.5, then the average is 3.5 ± 1.0, a value outside the range of the two measured values.

par_i	fitted X_i_____+-std.dev.	initial X_i_____+-std.dev	pull
1	4.50000 1.00000	4.50000 1.00000	
2	5.50000 1.00000	5.50000 1.00000	
3	-1.00000 0.210734E-07	0.00000 1.00000	
4	3.50000 1.00000	4.00000	

Now the second uncertainty, described by a full and positive correlation between the two values, is active, and moves the average value down. Although the problem is artificial and unrealistic, the result obtained by APCALC is sound. The uncertainty of the fitted variable x_3, describing the systematic uncertainty 2, approaches zero. In a more realistic scenario one would expect some uncorrelated uncertainty of the two measured values, which would change the problem and its solution considerably.

References

[1] J.M. Tinstra, *An Extension of the Technique of the Method of Least Squares to Correlated Observations*, Bulletin Geódésique Nr. 6 Paris (1947)

[2] Rudolf Boeck, *Application of a generalized method of least squares for kinematical analysis of tracks in bubble chambers*, CERN Yellow report 60–30 (1960)

[3] J. Peter Berge, Frank T. Solmitz and Horace D. Taft, *Kinematical Analysis of Interaction Vertices from Bubble Chamber Data*, Review of Scientific Instruments **32**, pp. 538 – 548 (1961)

[4] Fred James, MINUIT *Minimization Package reference manual*, CERN Program Library D505 (1994)

[5] Volker Blobel, *Constrained Least Squares and Error Propagation*, http://www.desy.de/~blobel/ (1997)

[6] Volker Blobel and Erich Lohrmann, *Statistische und numerische Methoden der Datenanalyse*, Teubner Studienbücher, Teubner (1998); e-book www.desy.de/~blobel/eBuch.pdf (2012)

[7] O. Bruno and D. Hoch, *Numerical Differentiation of Approximated Functions with Limited Order-of-Accuracy Determination*, SIAM J. Numer.Anal. 50, no. 5, pp. 1591–1603 (2012)

[8] Alfio Quarterioni, Riccardo Sacco and Fausto Saleri, *Numerical Mathematics*, Texts in Applied Mathematics, Springer (2007)

[9] C. Canuto, M.Hussaini, A. Quarteroni and T.A. Zang, *Spectral Methods: Fundamentals in Single Domains*, Springer (2006)

[10] A. Valassi, *Combining correlated measurements of several different physical quantities*, Nuclear Instruments and Methods A **500**, pp. 391–405 (2003)

[11] C.A. Cantrell, *Technical Note: Review of methods for least-squares fitting of data and application to atmospheric chemistry problems*, Atmos. Chem. Phys. **8**, 2008, pp. 5477 – 5487

[12] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P., *Numerical Recipes in C*, 2nd ed., Cambridge University Press, (1992)

[13] K. Pearson, *On lines and planes of closest fit to systems of points in space*, Philos. Mag., 2(2), 559–572 (1901)

[14] York, D., *Least-squares fitting of a straight line*, Can. J. Phys., 44, 1079–1086 (1966)

[15] D.S. Sivia, *Data analysis – a dialogue with the data*, Advanced Mathematical and Computational Tools in Metrology VII, World Scientific Publishing Co., pp. 108-118 (2006)

[16] Kenneth M. Hanson, Toshihiko Kawano, and Patrick Talou, *Probabilistic Interpretation of Peelle's Pertinent Puzzle and its Resolution*, Proc. Int. Conf. Nuclear Data for Science and Technology, AIP Conf. Proc. 769, pp. 304-307 (AIP, Melville, 2005)

[17] R. Frühwirth, D. Neudecker and H. Leeb, *Peelle's Pertinent Puzzle and its Solution*, EPJ Web of Conferences 27, 00008 (2012)

[18] R. W. Peelle, *Peelle's Pertinent Puzzle*, informal Oak Ridge National Laboratory memorandum, October 13, 1987.

[19] G. D'Agostini, *On the use of the covariance matrix to fit correlated data*, Nucl. Instrum. Methods. A346 (1994) 306.

[20] Thomas R. Junk, *Practical Issues in Statistical Interpretation of Tevatron Data*, Progress on Statistical Issues in Searches Conference, SLAC (2012)

[21] F.D. Aaron et al., *Combined Measuement and QCD Analysis of the Inclusive $e^{\pm}p$ Scattering Cross Sections at HERA*, Report DESY 09-158, JHEP **01**, p. 109, arxiv:0911.088 (2010)

[22] F.D. Aaron et al., *Measurement of the Inclusive ep Scattering Cross Section at Low Q^2 and x at HERA*, Eur.Phys.J.C **63**, p. 625 (2009) JHEP **01**, p. 109, arxiv:0904.0929 (2010)

[23] Frederick James, *Statistical Methods In Experimental Physics* (2Nd Edition), World Scientific (1006)

[24] John Nelder and Robert Wedderburn, *Generalized Linear Models*, Journal of the Royal Statistical Society. Series A (General) (Blackwell Publishing) **135 (3)** pp. 370–384 (1972)

[25] Peter McCullagh and John Nelder, *Generalized Linear Models*, Second Edition. Boca Raton: Chapman and Hall, (1989)

[26] Peter J. Huber, *Robust Statistics*, John Wiley (1981)

[27] D. Stump, J. Pumplin, R. Brock, D. Casey, J. Huston, J. Kalk, H.L. Lai and W.K. Tung, *Uncertainties of predictions from parton distribution functions. I. The Lagrange multiplier method*, Phys. Rev. D, Volume **65**, 014012 (2001)

[28] A.N. Tikhonov, *Solution of incorrectly formulated problems and the regularization method*, Soviet Math. Dokl., **4**, pp. 1035–1038 (1963); English translation of Dokl. Akad. Nauk. SSSR, **151**, pp. 501–504 (1963)

[29] Luc Demortier and Louis Lyons. *Everything you always wanted to know about pulls*, Note CDF/ANAL/PUBLIC/5776 (2002)

[30] Jorge J. Moré and Stefan M. Wild, *Estimating Computational Noise*, Argonne National Laboratory Preprint ANL/MCS-P1721-0210 (2011)