PEQN, PEQL - CGS Based Algorithms for Sparse Systems of Nonlinear Equations

Lukšan, Ladislav
1997

INSTITUTE OF COMPUTER SCIENCE

ACADEMY OF SCIENCES OF THE CZECH REPUBLIC

Prague

PEQN, PEQL - CGS Based Algorithms for Sparse Systems of Nonlinear Equations

L. Lukšan, J. Vlček

Technical Report No. V-719

September 1997

Akademie věd České republiky

ÚSTAV INFORMATIKY A VÝPOČETNÍ TECHNIKY

Institute of Computer Science, Academy of Sciences of the Czech Republic

Pod vodárenskou věží 2, 182 07 Prague 8, Czech Republic

E-mail: ICS@uivt.cas.cz

Fax: (+422) 8585789 Phone: (+422) 846669, (+422) 66051111

PEQN, PEQL - CGS Based Algorithms for Sparse Systems of Nonlinear Equations [1]

L. Lukšan and J. Vlček
Institute of Computer Science, Academy of Sciences of the Czech Republic,
Pod vodárenskou věží 2, 182 07 Prague 8, Czech Republic

Abstract. We present FORTRAN subroutines for solving sparse systems of nonlinear equations based on a doubly smoothed conjugate gradient squared method.

Categories and Subject Descriptors:

General Terms: Algorithms

Additional Key Words and Phrases: Nonlinear equations, sparse systems, discrete Newton method, limited memory quasi-Newton method, iterative solution of linear systems, conjugate gradient squared, residual smoothing

---

1. Introduction

The double-precision FORTRAN 77 basic subroutines PEQN and PEQL are designed to find a close approximation to a solution of the sparse system of nonlinear equations $f(x) = 0$. Here $x \in R^n$ is a vector of n variables and a mapping $f : R^n \to R^n$ is assumed to be continuously differentiable. Subroutine PEQN is based on a discrete (difference version) Newton method described in [2]. Subroutine PEQL is based on an inverse column update method proposed in [3] (see also [2]) which is in fact a limited memory quasi-Newton method. To simplify the user's work, two additional easy to use subroutines PEQNU and PEQLU are added, which call the basic general subroutines PEQN and PEQL. All subroutines contain a description of formal parameters and extensive comments. Furthermore, two test programs TEQNU and TEQLU are included, which contain 30 standard test problems (see [2]). These test programs serve as examples for using the subroutines, verify their correctness and demonstrate their efficiency.

2. The methods

Consider the system of nonlinear equations

$$f(x) = 0$$

[1] This work was supported under grant No. 201/96/0918 given by the Czech Republic Grant Agency

where, $f : R^n \to R^n$ is a continuously differentiable mapping. Let A be an approximation of the Jacobian matrix $J = J(x)$, where $J_{kl}(x) = \partial f_k(x)/\partial x_l$, $1 \le k \le n$, $1 \le l \le n$, and let $F = F(x) = (1/2)\|f(x)\|^2$. Algorithms presented in this paper belong to the class of Armijo-type descent methods, which generate the sequence of points $x_i \in R^n$, $i \in N$, such that

$$x_{i+1} = x_i + \alpha_i d_i, \quad i \in N.$$

Here $d_i \in R^n$ is the direction vector determined as an approximate solution of the linear system $A_i d + f_i = 0$ such that

$$\|A_i d_i + f_i\| \le \overline{\omega}_i \|f_i\|$$

with the precision $0 \le \overline{\omega}_i \le \overline{\omega} < 1$ and α_i is the stepsize obtained by the line search so that it is the first member of the sequence α_i^j, $j \in N$, where $\alpha_i^1 = 1$ and $\underline{\beta}\alpha_i^j \le \alpha_i^{j+1} \le \overline{\beta}\alpha_i^j$ with $0 < \underline{\beta} \le \overline{\beta} < 1$, satisfying

$$F_{i+1} - F_i \le -2\underline{\rho}(1 - \overline{\omega})\alpha_i F_i,$$

with the line search parameter $0 < \underline{\rho} < 1$ (parameter TOLS in the subroutines PEQN and PEQL). The value $\underline{\beta}\alpha_i^j \le \alpha_i^{j+1} \le \overline{\beta}\alpha_i^j$ can be determined either by the bisection (MES=1) or by two point quadratic interpolation (MES=2) or by three point quadratic interpolation (MES=3) or by three point cubic interpolation (MES=4) (MES is a parameter of the subroutines PEQN and PEQL).

To obtain a superlinear rate of convergence, the condition $\overline{\omega}_i \to 0$ have to be satisfied. Therefore, we choose $\overline{\omega}_i = \min(\max(\|f_i\|^\nu, \gamma(\|f_i\|/\|f_{i-1}\|)^\alpha), 1/i, \overline{\omega})$, with the values $\nu = 1/2$, $\gamma = 1$, $\alpha = (1 + \sqrt{5})/2$ and $\overline{\omega} = 1/2$.

If $A_i \ne J_i$, then a safeguard based on restarts is used. It consists in setting $A_{i+1} = J_{i+1}$ if $j > \underline{j}$ or $A_i = J_i$ (with repeating the i-th iteration) if $j > \overline{j}$, where $0 < \underline{j} < \overline{j}$. We use the values $\underline{j} = 1$ and $\overline{j} = 5$. The restart of the form $A_i = J_i$ is also used whenever $-d_i^T J_i^T f_i \le \underline{\varepsilon}\|d_i\|\|J_i^T f_i\|$, where $0 < \underline{\varepsilon} < 1$ is a tolerance for descent direction (parameter TOLD in the subroutines PEQN and PEQL).

The direction vector d_i (an approximate solution of the linear system $A_i d + f_i = 0$) is determined by the preconditioned smoothed CGS method which is defined by the following process. First we compute the vectors $s_1 = -C_i^{-1} f_i$, $r_1 = A_i s_i + f_i$. If $\|r_1\| \le \overline{\omega}\|f_i\|$, then we set $d_i = s_1$ and terminate the process. Otherwise we set $s_1 = 0$, $\overline{s}_1 = 0$, $r_1 = f_i$, $\overline{r}_1 = f_i$, $p_1 = f_i$, $u_1 = f_i$, $h_1 = A_i^T f_i$ and for $j = 1, 2, 3, \ldots$ we proceeds in the following way. If $\|r_j\| \le \overline{\omega}\|f_i\|$, then we set $d_i = s_j$ and terminate the process. Otherwise we compute

$$\begin{aligned}
v_j &= A_i C_i^{-1} p_j, \quad \alpha_j = h_i^T \overline{r}_j / h_i^T v_j, \\
q_j &= u_j - \alpha_j v_j, \\
\overline{s}_{j+1} &= \overline{s}_j + \alpha_j C_i^{-1}(u_j + q_j), \\
\overline{r}_{j+1} &= \overline{r}_j + \alpha_j A_i C_i^{-1}(u_j + q_j), \quad \beta_j = h_i^T \overline{r}_{j+1} / h_i^T \overline{r}_j,
\end{aligned}$$

$$
\begin{aligned}
u_{j+1} &= \overline{r}_{j+1} + \beta_j q_j, \\
p_{j+1} &= u_{j+1} + \beta_j (q_j + \beta_j p_j), \\
[\lambda_j, \mu_j]^T &= \arg \min_{[\lambda,\mu]^T \in \mathcal{R}^2} \|\overline{r}_{j+1} + \lambda(r_j - \overline{r}_{j+1}) + \mu v_j\|, \\
s_{j+1} &= \overline{s}_{j+1} + \lambda_j (s_j - \overline{s}_{j+1}) + \mu_j C_i^{-1} p_j, \\
r_{j+1} &= \overline{r}_{j+1} + \lambda_j (r_j - \overline{r}_{j+1}) + \mu_j v_j.
\end{aligned}
$$

The matrix C_i serves as a preconditioner. The choice $C_i = I$ is used if MOS2=1 or an incomplete LU decomposition of the matrix A_i is used if MOS2=2 (MOS2 is a parameter of the subroutines PEQN and PEQL).

More details concerning globally convergent Armijo type descent methods can be found in [2].

2.1 Discrete Newton method

The discrete Newton method is very simple. It is based on the elementwise differentiation. We always set $A = J$, where

$$
J_{kl}(x) = \frac{f_k(x + \delta_l e_l) - f_k(x)}{\delta_l},
$$

for all pairs (k, l) corresponding to structurally nonzero elements of $J(x)$. Thus we need m scalar function evaluations (i.e. m/n equivalent vector function evaluations) where m is the number of structurally nonzero elements of $J(x)$.

2.1 Inverse column update method

Inverse column update method, which was introduced in [3], uses an approximation $S = A^{-1}$ of the inverse Jacobian matrix $J^{-1}(x)$. Therefore, we simply set $s := -Sf$ instead of using the preconditioned smoothed CGS method if the restart was not used (if $A \neq J$). Denote by $d = x^+ - x$, $d_{-1} = x - x_{-1}$, ..., $d_{-k} = x_{1-k} - x_{-k}$ and $y = f(x^+) - f(x)$, $y_{-1} = f(x) - f(x_{-1})$, ..., $y_{-k} = f(x_{1-k}) - f(x_{-k})$ the last k differences of points and function vectors, respectively, where the lower index $-k$ corresponds to the iteration with the restart. Let $e_{-1} = \arg\max_{e_i} |e_i^T y_{-1}|$, ..., $e_{-k} = \arg\max_{e_i} |e_i^T y_{-k}|$ (arg max is taken over all e_i, $1 \leq i \leq n$). Then the vector Sf can be computed by the formula

$$
Sf = S_{-k} f + \frac{e_{-1}^T f}{e_{-1}^T y_{-1}} v_{-1} + \ldots + \frac{e_{-k}^T f}{e_{-k}^T y_{-k}} v_{-k},
$$

where $v_{-1} = d_{-1} - S_{-1} y_{-1}$, ..., $v_{-k} = d_{-k} - S_{-k} y_{-k}$ are vectors computed recursively by the formula

$$
Sy = S_{-k} y + \frac{e_{-1}^T y}{e_{-1}^T y_{-1}} v_{-1} + \ldots + \frac{e_{-k}^T y}{e_{-k}^T y_{-k}} v_{-k}.
$$

In both of these formulae we use the matrix $S_{-k} = (L_{-k} U_{-k})^{-1}$, where $L_{-k} U_{-k}$ is the incomplete LU decomposition of the Jacobian matrix $J(x_{-k})$. Note that the vectors e_{-1},

..., e_{k} do not need to be stored. We only use indices of their unique nonzero elements. The limited memory column update method needs to be restarted periodically after $\overline{k}$ iterations (parameter MF in the subroutine PEQL), since, at the most, $\overline{k}$ vectors can be stored.

3. Description of the subroutines

In this section we describe all subroutines which can be called from the user's programs. In the description of formal parameters we introduce a type of the argument that specifies whether the argument must have a value defined on entry to the subroutine (I), whether it is a value which will be returned (O), or both (U), or whether it is an auxiliary value (A). Besides formal parameters, we can use a COMMON /STAT/ block containing statistical information. This block, used in each subroutine, has the following form

```
COMMON /STAT/ NDECF,NRES,NRED,NREM,NADD,NIT,NFV,NFG,NFH
```

The arguments have the following meanings.

Argument	Type	Significance
NDECF	O	Positive INTEGER variable that indicates the number of matrix decompositions.
NRES	O	Positive INTEGER variable that indicates the number of restarts.
NRED	O	Positive INTEGER variable that indicates the number of reductions.
NREM	O	Positive INTEGER variable that indicates the number of constraint deletions during the QP solutions.
NADD	O	Positive INTEGER variable that indicates the number of constraint additions during the QP solutions.
NIT	O	Positive INTEGER variable that indicates the number of iterations.
NFV	O	Positive INTEGER variable that indicates the number of function evaluations.
NFG	O	Positive INTEGER variable that specifies the number of gradient evaluations.
NFH	O	Positive INTEGER variable that specifies the number of Hessian evaluations.

3.1 Subroutines PEQNU, PEQLU

The calling sequences are

```
CALL PEQNU(N,X,IAG,JAG,IA,RA,IPAR,RPAR,AF,F,GMAX,ITERM)

CALL PEQLU(N,X,IAG,JAG IA,RA,IPAR,RPAR,AF,F,GMAX,MF,ITERM)
```

The arguments have the following meaning.

Argument	Type	Significance
N	I	Positive INTEGER variable that specifies the number of equations and unknowns.
X(N)	U	On input, DOUBLE PRECISION vector with the initial estimate to the solution. On output, the approximation to the solution.
IAG(N+1)	I	INTEGER array containing row pointers of the sparse Jacobian matrix. More specifically, IAG(I)-1 is the number of structurally nonzero elements of the sparse Jacobian matrix in the rows with indices less than I.
JAG(M)	I	INTEGER array containing column indices of structurally nonzero elements of the Jacobian matrix (M=IAG(N+1)-1 is the number of structurally nonzero elements of the Jacobian matrix). These column indices are sorted rowwise by the increasing order.
IA(NIA)	A	INTEGER working array of the dimension NIA, where at last NIA=5*N for the subroutine PEQNU and at least NIA=6*N for the subroutine PEQLU.
RA(NRA)	A	DOUBLE PRECISION working array of the dimension NRA, where at least NRA=12*N+2*M for the subroutine PEQNU and at least NRA=(MF+12)*N+2*M+MF for the subroutine PEQLU (M=IAG(N+1)-1 is the number of nonzero elements of the Jacobian matrix).
IPAR(7)	A	INTEGER parameters. IPAR(1)=MES, IPAR(2)=MOS1, IPAR(3)=MOS2, IPAR(4)=MOS3, IPAR(5)=MIT, IPAR(6)=MFV, IPAR(7)=IPRNT. These parameters (MES, MOS1, MOS2, MOS3, MIT, MFV, IPRNT) are described in Section 3.2).
RPAR(8)	A	DOUBLE PRECISION parameters. RPAR(1)=TOLX, RPAR(2)=TOLF, RPAR(3)=TOLB, RPAR(4)=TOLG, RPAR(5)=TOLD, RPAR(6)=TOLS, RPAR(7)=ETA2, RPAR(8)=XMAX. These parameters (TOLX, TOLF, TOLB, TOLG, TOLD, TOLS, ETA2, XMAX) are described in Section 3.2).
AF(N)	O	DOUBLE PRECISION vector containing residuals of the nonlinear equations at the solution X.
F	O	DOUBLE PRECISION value of the objective function $(1/2)f^T f$ (sum of squares) at the solution X.

GMAX	O	DOUBLE PRECISION maximum absolute value of a partial derivative of the objective function.
MF	O	INTEGER variable that specifies the number of limited memory variable metric steps
ITERM	O	INTEGER variable that indicates the cause of termination: ITERM = 1: if $\|x - x_{old}\|$ was less than or equal to TOLX in MTESX subsequent iterations, ITERM = 2: if $\|F - F_{old}\|$ was less than or equal to TOLF in MTESF subsequent iterations, ITERM = 3: if F is less than or equal to TOLB, ITERM = 4: if GMAX is less than or equal to TOLG, ITERM = 11: if NFV exceeded MFV, ITERM = 12: if NIT exceeded MIT, ITERM < 0: if the method failed.

The subroutines PEQNU, PEQLU require the user supplied subroutine FUN that defines the values of the elements of the mapping $f : R^n \to R^n$ (residuals of the nonlinear equations) and has the form

```
SUBROUTINE FUN(NF,KA,X,FA)
```

The arguments of the user supplied subroutine have the following meaning.

Argument	Type	Significance
N	I	Positive INTEGER variable that specifies the number of equations and unknowns.
KA	I	Positive INTEGER variable that specifies the index of the equation.
X(N)	I	DOUBLE PRECISION an estimate to the solution.
FA	O	DOUBLE PRECISION value of the residual with the index KA at the point X.

3.2 Subroutines PEQN, PEQL

The general subroutine PEQN is called from the subroutine PEQU described in Section 3.1. The calling sequence is

```
 CALL PEQN(N,X,GA,AG,IAG,JAG,IB,IW1,IW2,IW3,IW4,G,S,XO,GO,XS,GS,
& XP,GP,AF,AFO,AFD,SO,TOLX,TOLF,TOLB,TOLG,TOLD,TOLS,ETA2,XMAX,
& ETA2,XMAX,GMAX,F,MES,MOS1,MOS2,MOS3,MIT,MFV,IPRNT,ITERM)
```

The general subroutine PEQL is called from the subroutine PEQLU described in Section 3.1. The calling sequence is

```
 CALL PEQL(N,X,GA,AG,IAG,JAG,IB,IW1,IW2,IW3,IW4,XM,GM,IM,G,S,
& XO,GO,XS,GS,XP,GP,AF,AFO,AFD,SO,TOLX,TOLF,TOLB,TOLG,TOLD,TOLS,
& ETA2,XMAX,GMAX,F,MES,MOS1,MOS2,MOS3,MIT,MFV,IPRNT,MF,ITERM)
```

The arguments N, X, IAG, JAG, AF, GMAX, F, MF, ITERM have the same meaning as in Section 3.1. Other arguments have the following meanings.

Argument	Type	Significance
GA(N)	A	DOUBLE PRECISION gradient of the selected residual (selected row of the Jacobian matrix).
AG(M)	A	DOUBLE PRECISION sparse Jacobian matrix (M=IAG(N+1)-1 is the number of nonzero elements in the Jacobian matrix)
IB(N)	A	INTEGER permutation vector.
IW1(N)	A	INTEGER auxiliary vector.
IW2(N)	A	INTEGER auxiliary vector.
IW3(N)	A	INTEGER auxiliary vector.
IW4(N)	A	INTEGER auxiliary vector.
XM(N*MF)	A	DOUBLE PRECISION set of vectors for an inverse column update.
GM(MF)	A	DOUBLE PRECISION set of values for an inverse column update.
IM(MF)	A	INTEGER set of indices for an inverse column update.
G(N)	A	DOUBLE PRECISION gradient of the objective function.
S(N)	A	DOUBLE PRECISION direction vector.
XO(N)	A	DOUBLE PRECISION auxiliary vector.
GO(N)	A	DOUBLE PRECISION auxiliary vector.
XS(N)	A	DOUBLE PRECISION auxiliary vector.
GS(N)	A	DOUBLE PRECISION auxiliary vector.
XP(N)	A	DOUBLE PRECISION auxiliary vector.
GP(N)	A	DOUBLE PRECISION auxiliary vector.
AFO(NA)	A	DOUBLE PRECISION auxiliary vector.
AFD(NA)	A	DOUBLE PRECISION auxiliary vector.
SO(NF)	A	DOUBLE PRECISION auxiliary vector.

TOLX	I	DOUBLE PRECISION tolerance for the change of the coordinate vector X; the choice TOLX = 0 causes that the default value 10^{-16} will be taken.
TOLF	I	DOUBLE PRECISION tolerance for the change of function values; the choice TOLF = 0 causes that the default value 10^{-16} will be taken.
TOLB	I	DOUBLE PRECISION minimum acceptable function value; the choice TOLB = 0 causes that the default value 10^{-16} will be taken.
TOLG	I	DOUBLE PRECISION tolerance for the gradient of the Lagrangian function; the choice TOLG = 0 causes that the default value 10^{-6} will be taken.
TOLD	I	DOUBLE PRECISION tolerance for descent direction; the choice TOLD = 0 causes that the default value 10^{-15} will be taken.
TOLS	I	DOUBLE PRECISION tolerance parameter for a function decrease in the line search; the choice TOLS = 0 causes that the default value 10^{-4} will be taken.
ETA2	I	DOUBLE PRECISION damping parameter for an incomplete LU preconditioner; the choice ETA2=0 is recommended in general.
XMAX	I	DOUBLE PRECISION maximum stepsize; the choice XMAX = 0 causes that the default value 10^5 will be taken.
MES	I	INTEGER variable that specifies the interpolation method selection in the line search: MES = 1: bisection, MES = 2: two point quadratic interpolation. MES = 3: three point quadratic interpolation. MES = 4: three point cubic interpolation. The choice MES = 0 causes that the default value MES = 1 will be taken.
MOS1	I	INTEGER choice of the dual vector in the CGS method. MOS1 = 1: gradient of the objective function, MOS1 = 2: vector of residuals of the nonlinear equations. The choice MOS1 = 0 causes that the default value MOS1 = 1 will be taken.
MOS2	I	INTEGER choice of the preconditioner. MOS2 = 1: no preconditioning, MOS2 = 2: incomplete LU decomposition as a preconditioner. The choice MOS2 = 0 causes that the default value MOS2 = 2 will be taken.
MOS3	I	INTEGER choice of smoothing the preconditioner.

MOS3 = 1: no smoothing,
MOS3 = 2: single smoothing strategy.
MOS3 = 3: double smoothing strategy.
The choice MOS3 = 0 causes that the default value MOS3 = 3 will be taken.

MIT I INTEGER variable that specifies the maximum number of iterations; the choice MIT = 0 causes that the default value 200 will be taken.

MFV I INTEGER variable that specifies the maximum number of function evaluations; the choice MFV = 0 causes that the default value 500 will be taken.

IPRNT I INTEGER variable that specifies PRINT:
IPRNT = 0: print is suppressed,
IPRNT = 1: basic print of final results,
IPRNT = −1: extended print of final results,
IPRNT = 2: basic print of intermediate and final results,
IPRNT = −2: extended print of intermediate and final results,

Subroutines PEQN and PEQL have a modular structure. The following list contains their most important subroutines.

PAOSQ3 Evaluation of the objective function (sum of squares).

PAOGS3 Numerical differentiation (determination of the Jacobian matrix).

PDSLE3 Determination of the direction vector using the preconditioned smoothed CGS method.

PDLLI3 Determination of the direction vector using a combination of the preconditioned smoothed CGS method and the limited memory inverse column update method.

PS0LB2 Line search using only function values.

PULCI3 Limited memory inverse column update.

Subroutines PEQN, PEQL require the user supplied subroutine FUN. User supplied subroutine FUN is described in Section 3.1.

3.3 Form of printed results

The form of printed results is specified by the parameter IPRNT as it is described above. Here we demonstrate individual forms of printed results by the simple use of the program TEQNU described in the next section (with NEXT=1 and N=50). If we set IPRNT=1, then the printed results will have the form

```
NIT=   9  NFV=  45  NFG=   0  F=  .37041508D-23  G= .0000D+00  ITERM=  3
```

If we set IPRNT=-1, then the printed results will have the form

```
EXIT FROM PEQN :
NIT=   9  NFV=  45  NFG=   0  F=  .37041508D-23  G= .0000D+00  ITERM=  3
X=  .5516874D+00 -.5555235D-01  .1418055D+00  .1187617D+00  .1333481D+00
   -.1426014D+00  .2723421D-01  .1853631D+00  .3849394D-01 -.1845711D+00
    .7085278D-02  .2037814D+00  .1277260D-01 -.2012282D+00  .2101740D-02
    .2089332D+00  .5056134D-02 -.2075359D+00  .3841057D-03  .2107999D+00
    .3640168D-02 -.2099278D+00 -.7828511D-03  .2122563D+00  .6535190D-02
   -.2110370D+00 -.2819896D-02  .2151212D+00  .1702794D-01 -.2121421D+00
   -.7977026D-02  .2227681D+00  .4719821D-01 -.2144874D+00 -.2138609D-01
    .2442434D+00  .1317577D+00 -.2203956D+00 -.5259093D-01  .3057817D+00
    .3655903D+00 -.2344361D+00 -.9811103D-01  .4867714D+00  .9438735D+00
   -.2753059D+00  .9297379D-01  .1038738D+01 -.1467769D-01 -.1041795D+01
```

If we set IPRNT=2, then the printed results will have the form

```
ENTRY TO PEQN :
NIT=   0  NFV=   1  NFG=   0  F=  .23371300D+02  G= .0000D+00
NIT=   1  NFV=   5  NFG=   0  F=  .45923924D+01  G= .0000D+00
NIT=   2  NFV=  10  NFG=   0  F=  .43038176D+01  G= .0000D+00
NIT=   3  NFV=  15  NFG=   0  F=  .27445445D+00  G= .0000D+00
NIT=   4  NFV=  20  NFG=   0  F=  .81827010D-02  G= .0000D+00
NIT=   5  NFV=  25  NFG=   0  F=  .22560535D-03  G= .0000D+00
NIT=   6  NFV=  30  NFG=   0  F=  .48710984D-06  G= .0000D+00
NIT=   7  NFV=  35  NFG=   0  F=  .76018472D-10  G= .0000D+00
NIT=   8  NFV=  40  NFG=   0  F=  .85908290D-15  G= .0000D+00
NIT=   9  NFV=  45  NFG=   0  F=  .37041508D-23  G= .0000D+00
EXIT FROM PEQN :
NIT=   9  NFV=  45  NFG=   0  F=  .37041508D-23  G= .0000D+00  ITERM=  3
```

If we set IPRNT=-2, then the printed results will have the form

```
ENTRY TO PEQN :
NIT=   0  NFV=   1  NFG=   0  F=  .23371300D+02  G= .0000D+00
NIT=   1  NFV=   5  NFG=   0  F=  .45923924D+01  G= .0000D+00
NIT=   2  NFV=  10  NFG=   0  F=  .43038176D+01  G= .0000D+00
NIT=   3  NFV=  15  NFG=   0  F=  .27445445D+00  G= .0000D+00
NIT=   4  NFV=  20  NFG=   0  F=  .81827010D-02  G= .0000D+00
NIT=   5  NFV=  25  NFG=   0  F=  .22560535D-03  G= .0000D+00
NIT=   6  NFV=  30  NFG=   0  F=  .48710984D-06  G= .0000D+00
NIT=   7  NFV=  35  NFG=   0  F=  .76018472D-10  G= .0000D+00
NIT=   8  NFV=  40  NFG=   0  F=  .85908290D-15  G= .0000D+00
NIT=   9  NFV=  45  NFG=   0  F=  .37041508D-23  G= .0000D+00
EXIT FROM PEQN :
```

```
NIT=   9  NFV=  45  NFG=   0  F=  .37041508D-23  G= .0000D+00  ITERM=  3
X=  .5516874D+00 -.5555235D-01  .1418055D+00  .1187617D+00  .1333481D+00
   -.1426014D+00  .2723421D-01  .1853631D+00  .3849394D-01 -.1845711D+00
    .7085278D-02  .2037814D+00  .1277260D-01 -.2012282D+00  .2101740D-02
    .2089332D+00  .5056134D-02 -.2075359D+00  .3841057D-03  .2107999D+00
    .3640168D-02 -.2099278D+00 -.7828511D-03  .2122563D+00  .6535190D-02
   -.2110370D+00 -.2819896D-02  .2151212D+00  .1702794D-01 -.2121421D+00
   -.7977026D-02  .2227681D+00  .4719821D-01 -.2144874D+00 -.2138609D-01
    .2442434D+00  .1317577D+00 -.2203956D+00 -.5259093D-01  .3057817D+00
    .3655903D+00 -.2344361D+00 -.9811103D-01  .4867714D+00  .9438735D+00
   -.2753059D+00  .9297379D-01  .1038738D+01 -.1467769D-01 -.1041795D+01
```

4. Verification of the subroutines

In this section we introduce the main programs TEQNU and TEQLU, which serve for demonstration, verification and testing the subroutines PEQNU and PEQLU.

4.1 Program TEQNU

The following main program demonstrates the usage of the subroutine PEQNU.

```
C
C     TEST PROGRAM FOR THE SUBROUTINE PEQNU
C
      INTEGER N,MM,IAG(101),JAG(2000),IA(500),IPAR(7),ITERM
      REAL*8 X(100),AF(100),RA(6000),RPAR(8),F,GMAX
      REAL*8 FMIN,PAR
      INTEGER NEXT,IERR,I,M
      COMMON /PROB/ NEXT
      INTEGER NDECF,NRES,NRED,NREM,NADD,NIT,NFV,NFG,NFH
      COMMON /STAT/ NDECF,NRES,NRED,NREM,NADD,NIT,NFV,NFG,NFH
C
C     COMMON /EMPARM/ IS USED ONLY IN TEST SUBROUTINES FROM THE
C     UFO SYSTEM (TIUB18, TAFU18)
C
      COMMON /EMPARM/ PAR,M
C
C     LOOP FOR 30 TEST PROBLEMS
C
      DO 3 NEXT=1,30
C
C     CHOICE OF INTEGER AND REAL PARAMETERS
C
      DO 1 I=1,7
```

```
      IPAR(I)=0
    1 CONTINUE
      DO 2 I=1,7
      RPAR(I)=0.0D 0
    2 CONTINUE
      IPAR(7)=1
C
C     PROBLEM DIMENSION
C
      N=100
C
C     INITIATION OF X, DETERMINATION IAG AND JAG AND CHOICE OF RPAR(8)
C
      CALL TIUB18(N,N,MM,X,IAG,JAG,FMIN,RPAR(8),NEXT,IERR)
      IF (NEXT.EQ.23) PAR=5.0D 1
      IF (NEXT.EQ.24) PAR=6.7D 0/DBLE(M+1)**2
      IF (NEXT.EQ.29) RPAR(7)=1.0D-2
      IF (NEXT.EQ.30) RPAR(7)=1.0D-2
      IF (IERR.NE.0) GO TO 3
C
C     SOLUTION
C
      CALL PEQNU(N,X,IAG,JAG,IA,RA,IPAR,RPAR,AF,F,GMAX,ITERM)
    3 CONTINUE
      STOP
      END
C
C     USER SUPPLIED SUBROUTINE (CALCULATION OF FA)
C
      SUBROUTINE FUN(NF,KA,X,FA)
      INTEGER NF,KA
      REAL*8 X(*),FA
      INTEGER NEXT
      COMMON /PROB/ NEXT
C
C     FUNCTION EVALUATION
C
      CALL TAFU18(NF,KA,X,FA,NEXT)
      RETURN
      END
```

This main program uses subroutines TIUB18 (initiation) and TAFU18 (function evaluation) containing 30 standard test problems with an arbitrary number of variables, which were taken from the UFO system [1]. Results obtained by this main program

have the following form.

```
NIT=  10  NFV=  53  NFG=   0  F=  .22299526D-23  G= .0000D+00  ITERM=  3
NIT=   8  NFV=  50  NFG=   0  F=  .39380444D-16  G= .0000D+00  ITERM=  3
NIT=   3  NFV=  19  NFG=   0  F=  .23549194D-19  G= .0000D+00  ITERM=  3
NIT=   7  NFV=  29  NFG=   0  F=  .34807130D-17  G= .0000D+00  ITERM=  3
NIT=   5  NFV=  30  NFG=   0  F=  .11101088D-16  G= .0000D+00  ITERM=  3
NIT=  16  NFV=  64  NFG=   0  F=  .11495745D-16  G= .0000D+00  ITERM=  3
NIT=  28  NFV= 125  NFG=   0  F=  .34936677D-27  G= .0000D+00  ITERM=  3
NIT=  12  NFV=  85  NFG=   0  F=  .12029898D-19  G= .0000D+00  ITERM=  3
NIT=  13  NFV= 106  NFG=   0  F=  .25199766D-20  G= .0000D+00  ITERM=  3
NIT=   5  NFV=  45  NFG=   0  F=  .12006685D-25  G= .0000D+00  ITERM=  3
NIT=  12  NFV=  37  NFG=   0  F=  .18971583D-17  G= .0000D+00  ITERM=  3
NIT=  17  NFV=  52  NFG=   0  F=  .68541487D-17  G= .0000D+00  ITERM=  3
NIT=  16  NFV=  47  NFG=   0  F=  .89255086D-16  G= .0000D+00  ITERM=  3
NIT=   4  NFV=  16  NFG=   0  F=  .22140523D-21  G= .0000D+00  ITERM=  3
NIT=   5  NFV=  40  NFG=   0  F=  .18251617D-17  G= .0000D+00  ITERM=  3
NIT=  53  NFV= 319  NFG=   0  F=  .15406006D-18  G= .0000D+00  ITERM=  3
NIT=  14  NFV=  55  NFG=   0  F=  .13775393D-23  G= .0000D+00  ITERM=  3
NIT=  15  NFV=  60  NFG=   0  F=  .23644102D-24  G= .0000D+00  ITERM=  3
NIT=   2  NFV=   8  NFG=   0  F=  .57689461D-18  G= .0000D+00  ITERM=  3
NIT=   9  NFV=  39  NFG=   0  F=  .34744489D-16  G= .0000D+00  ITERM=  3
NIT=  12  NFV=  48  NFG=   0  F=  .52954307D-16  G= .0000D+00  ITERM=  3
NIT=   7  NFV=  59  NFG=   0  F=  .18591459D-23  G= .0000D+00  ITERM=  3
NIT=   7  NFV=  56  NFG=   0  F=  .29518455D-25  G= .0000D+00  ITERM=  3
NIT=   6  NFV=  34  NFG=   0  F=  .55206888D-22  G= .0000D+00  ITERM=  3
NIT=   6  NFV=  34  NFG=   0  F=  .33126463D-17  G= .0000D+00  ITERM=  3
NIT=  11  NFV=  62  NFG=   0  F=  .34795933D-18  G= .0000D+00  ITERM=  3
NIT=   9  NFV=  51  NFG=   0  F=  .11531451D-21  G= .0000D+00  ITERM=  3
NIT=  10  NFV=  69  NFG=   0  F=  .68878264D-17  G= .0000D+00  ITERM=  3
NIT=   5  NFV=  61  NFG=   0  F=  .46977618D-19  G= .0000D+00  ITERM=  3
NIT=   9  NFV= 114  NFG=   0  F=  .55194572D-18  G= .0000D+00  ITERM=  3
```

The rows corresponding to individual test problems contain the number of iterations NIT, the number of function evaluations NFV, the number of gradient evaluations NFG, the final value of the objective function F, the final gradient norm G and the cause of termination ITERM.

4.2 Program TEQLU

The following main program demonstrates the usage of the subroutine PEQLU.

```
C
C     TEST PROGRAM FOR THE SUBROUTINE PEQLU
C
```

```
      INTEGER N,MM,MF,IAG(101),JAG(2000),IA(600),IPAR(7),ITERM
      REAL*8 X(100),AF(100),RA(6000),RPAR(8),F,GMAX
      REAL*8 FMIN,PAR
      INTEGER NEXT,IERR,I,M
      COMMON /PROB/ NEXT
      INTEGER NDECF,NRES,NRED,NREM,NADD,NIT,NFV,NFG,NFH
      COMMON /STAT/ NDECF,NRES,NRED,NREM,NADD,NIT,NFV,NFG,NFH
C
C     COMMON /EMPARM/ IS USED ONLY IN TEST SUBROUTINES FROM THE
C     UFO SYSTEM (TIUB18, TAFU18)
C
      COMMON /EMPARM/ PAR,M
C
C     LOOP FOR 30 TEST PROBLEMS
C
      DO 3 NEXT=1,30
C
C     CHOICE OF INTEGER AND REAL PARAMETERS
C
      DO 1 I=1,7
      IPAR(I)=0
    1 CONTINUE
      DO 2 I=1,7
      RPAR(I)=0.0D 0
    2 CONTINUE
      IPAR(7)=1
C
C     PROBLEM DIMENSION
C
      N=100
C
C     NUMBER OF STEPS OF THE LIMITED MEMORY INVERSE COLUMN UPDATING
C     METHOD
C
      MF=6
C
C     INITIATION OF X, DETERMINATION IAG AND JAG AND CHOICE OF RPAR(8)
C
      CALL TIUB18(N,N,MM,X,IAG,JAG,FMIN,RPAR(8),NEXT,IERR)
      IF (NEXT.EQ.23) PAR=5.0D 1
      IF (NEXT.EQ.24) PAR=6.7D 0/DBLE(M+1)**2
      IF (NEXT.EQ.29) RPAR(7)=1.0D-2
      IF (NEXT.EQ.30) RPAR(7)=1.0D-2
```

```
      IF (IERR.NE.0) GO TO 3
C
C     SOLUTION
C
      CALL PEQLU(N,X,IAG,JAG,IA,RA,IPAR,RPAR,AF,F,GMAX,MF,ITERM)
    3 CONTINUE
      STOP
      END
C
C     USER SUPPLIED SUBROUTINE (CALCULATION OF FA)
C
      SUBROUTINE FUN(NF,KA,X,FA)
      INTEGER NF,KA
      REAL*8 X(*),FA
      INTEGER NEXT
      COMMON /PROB/ NEXT
C
C     FUNCTION EVALUATION
C
      CALL TAFU18(NF,KA,X,FA,NEXT)
      RETURN
      END
```

This main program uses subroutines TIUB18 (initiation) and TAFU18 (function evaluation) containing 30 standard test problems with an arbitrary number of variables, which were taken from the UFO system [1]. Results obtained by this main program have the following form.

```
NIT=  15  NFV=  39  NFG=   0  F=  .20662478D-18  G= .0000D+00  ITERM=  3
NIT=  10  NFV=  61  NFG=   0  F=  .20794489D-19  G= .0000D+00  ITERM=  3
NIT=   4  NFV=  15  NFG=   0  F=  .24002496D-20  G= .0000D+00  ITERM=  3
NIT=  10  NFV=  23  NFG=   0  F=  .21132659D-20  G= .0000D+00  ITERM=  3
NIT=   8  NFV=  25  NFG=   0  F=  .12302291D-17  G= .0000D+00  ITERM=  3
NIT=  17  NFV=  28  NFG=   0  F=  .66270979D-16  G= .0000D+00  ITERM=  3
NIT=  60  NFV= 352  NFG=   0  F=  .12488383D-25  G= .0000D+00  ITERM=  3
NIT=  20  NFV=  83  NFG=   0  F=  .43111659D-23  G= .0000D+00  ITERM=  3
NIT=  21  NFV=  82  NFG=   0  F=  .36170451D-28  G= .0000D+00  ITERM=  3
NIT=   7  NFV=  31  NFG=   0  F=  .62633189D-25  G= .0000D+00  ITERM=  3
NIT=  16  NFV=  25  NFG=   0  F=  .19721523D-26  G= .0000D+00  ITERM=  3
NIT=  18  NFV=  29  NFG=   0  F=  .47126179D-16  G= .0000D+00  ITERM=  3
NIT=  20  NFV=  31  NFG=   0  F=  .61842712D-16  G= .0000D+00  ITERM=  3
NIT=   5  NFV=  14  NFG=   0  F=  .66660487D-24  G= .0000D+00  ITERM=  3
NIT=   7  NFV=  28  NFG=   0  F=  .81081987D-16  G= .0000D+00  ITERM=  3
NIT=  37  NFV= 185  NFG=   0  F=  .75750824D-23  G= .0000D+00  ITERM=  3
```

```
NIT=  14  NFV=  41  NFG=   0  F=  .79197522D-17  G= .0000D+00  ITERM=  3
NIT=  18  NFV=  46  NFG=   0  F=  .35202092D-21  G= .0000D+00  ITERM=  3
NIT=   2  NFV=   8  NFG=   0  F=  .57689461D-18  G= .0000D+00  ITERM=  3
NIT=  14  NFV=  29  NFG=   0  F=  .22957080D-24  G= .0000D+00  ITERM=  3
NIT=  23  NFV=  35  NFG=   0  F=  .25221528D-19  G= .0000D+00  ITERM=  3
NIT=   9  NFV=  51  NFG=   0  F=  .16004490D-17  G= .0000D+00  ITERM=  3
NIT=  12  NFV=  48  NFG=   0  F=  .24806709D-20  G= .0000D+00  ITERM=  3
NIT=   9  NFV=  37  NFG=   0  F=  .71989727D-18  G= .0000D+00  ITERM=  3
NIT=  10  NFV=  29  NFG=   0  F=  .33369952D-18  G= .0000D+00  ITERM=  3
NIT=  17  NFV=  64  NFG=   0  F=  .20817177D-18  G= .0000D+00  ITERM=  3
NIT=  18  NFV=  42  NFG=   0  F=  .14034691D-16  G= .0000D+00  ITERM=  3
NIT=  12  NFV=  72  NFG=   0  F=  .52517008D-17  G= .0000D+00  ITERM=  3
NIT=   9  NFV=  66  NFG=   0  F=  .15213964D-21  G= .0000D+00  ITERM=  3
NIT=  17  NFV= 133  NFG=   0  F=  .57811565D-17  G= .0000D+00  ITERM=  3
```

The rows corresponding to individual test problems contain the number of iterations NIT, the number of function evaluations NFV, the number of gradient evaluations NFG, the final value of the objective function F, the final gradient norm G and the cause of termination ITERM.

References

[1] Lukšan L., Šiška M., Tůma M., Vlček J., Ramešová N. Interactive System for Universal Functional Optimization (UFO), Version 1996. Research Report No. V-701, Institute of Computer Science, Academy of Sciences of the Czech Republic, Prague, Czech Republic, 1996.

[2] Lukšan L., Vlček J. Computational Experience with Globally Convergent Descent Methods for Large Sparse Systems of Nonlinear Equations. To appear in Optimization Methods and Software.

[3] Martinez J.M., Zambaldi M.C. An Inverse Column-Updating Method for Solving Large-Scale Nonlinear Systems of Equations. Optimization Methods and Software 1 (1992) 129-140.