An Assessment of the Accuracy of the Least Squares Procedures of Several Computer Software Packages

Harald HEINZL

Forschungsbericht/
Research Memorandum No. 282

June 1991

The author is very grateful to R. W. FAREBROTHER (visiting professor at the Institute for Advanced Studies in July 1990) for his constructive comments.

Zusammenfassung: Der Datensatz von Longley dient als Grundlage für die folgenden Untersuchungen: Erstens, die Bewertung der numerischen Genauigkeit der in verschiedenen Computerprogrammpaketen zur Verfügung stehenden OLS-Prozeduren. Zweitens, die empirische Berechnung der Einzel-Limes-Konditionalzahl von Farebrother. Und schließlich, die Einschätzung der numerischen Effekte von wiederholten Matrizeninversionen. Eines der wichtigsten Resultate dieser Studie ist die im allgemeinen deutliche Verbesserung der numerischen Genauigkeit der OLS-Ergebnisse, welche sich nach einer spaltenweisen Mittelwertzentrierung der zugrundeliegenden Datenmatrix ergibt.

Schlüsselwörter: Numerische Genauigkeit, OLS-Prozedur, Longley's Testproblem, Konditionalzahl.

Abstract: Longley's data set is used as a basis for the following projects: Firstly, to assess the numerical accuracy of the least squares procedures of several computer software packages. Secondly, to calculate Farebrother's single limit condition number empirically. Finally, to assess the numerical effect of repeated matrix inversions. It mainly appears that mean-centering of the variables can substantially improve the numerical accuracy and stability of the computed OLS-estimates.

Key words: Numerical accuracy, OLS-procedure, Longley's test problem, condition number.

Contents

1. Introduction

Today, almost everyone uses computer software, if an arithmetical problem arises. But only few people ask themselves how accurate the solutions are, which they get from their computers.

Most people trust in their electronical machines like in God and forget two important facts:

- Due to creation by humans each software product can have (and often actually does have) more or less severe programming bugs.

- Due to hardware restrictions each software product can only use limited space to handle and to store numbers.

Let's suppose for a moment, we have a perfect, bug-free program, which stores both real and integer numbers as strings of 32 bits. So there are 2^{32} distinct strings of 32 binary digits, which may be used either to represent the integers from -2^{31} to $2^{31}-1$ or to represent real numbers in floating point form, $a \cdot 2^b$.

Let's for example reserve for the mantissa a 24 digits and for the exponent b 8 digits (with 1 digit reserved for the sign in both), then positive real numbers between $1.5 \cdot 10^{-39}$ and $1.7 \cdot 10^{38}$, and negative real numbers between $-1.7 \cdot 10^{38}$ and $-1.5 \cdot 10^{-39}$ may be represented.

The accuracy is 24 binary digits[1] or 7.22 decimal digits ($2^{24} = 10^{7.22}$). As a rule of thumb we can say:

- Each byte (8 bits) mantissa length brings 2.41 decimal digits more accuracy.

- One decimal digit more accuracy needs 3.32 bits more mantissa length.

---

1) 23 bits in the mantissa without the sign-bit, plus 1 digit which results from the fact that each dual number begins with a "1" - so this leading digit can be omitted without any loss of information. See also Farebrother (1988), pp. 15-16.

We see, even a bug-free program produces incorrect solutions. The crucial questions are:

- What amount of error must we expect, if we run a specific program?

- What can we do to minimize this incorrectness?

This paper concerns with these questions in context with several programs for OLS – estimation, which are used at the Institute for Advanced Studies in Vienna.

The following software packages have been examined[2]:

1.) IAS-System (Inter-active Simulation System, Level IAS-3.8.0)

2.) RATS (Regression Analysis of Time Series, Version 3.03)

3.) GAUSS (The GAUSS System, Version 2.0)

4.) 1-2-3 (LOTUS 1-2-3, Version 3.1)

5.) EXCEL (Microsoft Excel, Version 2.1c)

6.) ASEASY (AS-EASY-AS, Versions 3.00 and 4.00b)

7.) MathCAD (MathCAD 2.50)

8.) MATRIX OPERATION-facility at the SHARP Pocket Computer PC-1403

---

2) At the Institute for Advanced Studies the first program package runs both on a UNISYS mainframe computer and on IBM-PC clones, the program packages 2 to 7 are implemented on IBM-PC clones and the last one (which isn't really a program *package*) is a built-in-facility at a pocket calculator. Each of the IBM-PC clones contains a 80386 processor and a 80387 coprocessor, the operating system in use is MS-DOS.

In order to test the OLS-estimation accuracy of these software products Longley's notorious data set[3] has been used, see tables[4] 1.1 and 1.2.

X1	X2	X3	X4	X5	X6	X7	Y
1	83.0	234289	2356	1590	107608	1947	60323
1	88.5	259426	2325	1456	108632	1948	61122
1	88.2	258054	3682	1616	109773	1949	60171
1	89.5	284599	3351	1650	110929	1950	61187
1	96.2	328975	2099	3099	112075	1951	63221
1	98.1	346999	1932	3594	113270	1952	63639
1	99.0	365385	1870	3547	115094	1953	64989
1	100.0	363112	3578	3350	116219	1954	63761
1	101.2	397469	2904	3048	117388	1955	66019
1	104.6	419180	2822	2857	118734	1956	67857
1	108.4	442769	2936	2798	120445	1957	68169
1	110.8	444546	4681	2637	121950	1958	66513
1	112.6	482704	3813	2552	123366	1959	68655
1	114.2	502601	3931	2514	125368	1960	69564
1	115.7	518173	4806	2572	127852	1961	69331
1	116.9	554894	4007	2827	130081	1962	70551

Table 1.1: Longley's data set

OLS-coefficients	
b1	-3482258.63459581833
b2	15.06187227137329497
b3	-0.0358191792925910166
b4	-2.02022980381682509
b5	-1.03322686717359198
b6	-0.0511041056535807145
b7	1829.15146461355185

Table 1.2: Longley's correct OLS-solution (18 digits accuracy)

3) See Longley (1967). It should be pointed out that Longley's data set is only the most famous of a lot of possible and meaningful ones, see e. g. Wampler (1970).

4) Reprinted from Farebrother (1988), pp. 52-53.

The variables of Longley's data set are highly correlated, which results in a very ill-conditioned X-matrix, which in consequence should help us to separate the good, i. e. the numerically stable programs from the bad ones.

Behind this approach there is the following assumption:

> If the data matrix is well-conditioned, then the numerical performance of the several OLS-programs will be expected to differ only slightly. But if the data matrix becomes more and more ill-conditioned, then these differences should become more and more bigger, since the decline in numerical accuracy of a stable program should be smaller than the decline of a bad one. Consequently one can expect that the use of a very ill-conditioned X-matrix is similar to the use of a very strong magnifier.

There is some evidence, that this assumption is a realistic one.

2. A Brief Description of the Examined Software Packages in Regard of Their Least Squares Facilities

2.1. IAS-System

In the IAS-System the ***OLS**-command performs OLS-estimation. We use the IAS-System in two different adaptations (to an UNISYS mainframe computer and to a PC).

With special commands up to 20 (mainframe) or 7 (PC) decimal digits of the results can be displayed.

2.2. RATS

In RATS there are two possibilities to perform OLS-estimation, the **LINREG**-command and the **MATRIX**-command, *MATRIX b=INV(TR(X)*X)*TR(X)*y*.

With special commands up to 16 decimal digits of the results can be displayed. RATS stores all data as double precision numbers (8 bytes per item).

2.3. GAUSS

In GAUSS there are at least seven more or less different possibilities to perform OLS-estimation.

1.) **OLS**-command, OLS uses the Cholesky decomposition

2.) **OLSQR**-command, QR stands for QR decomposition

3.) /-operator, variant I, $b=X'y/X'X$, this variant is based on LU decomposition

4.) /-operator, variant II, $b=y/X$, this variant is based on forming the normal equations and using the Cholesky decomposition to get the solution

5.) **INV**-command[5], $b=INV(X'*X)*X'*y$, INV inverts invertible matrices, it uses the Crout decomposition

6.) **INVPD**-command[6], $b=INVPD(X'*X)*X'*y$, INVPD inverts only symmetric, positive definite matrices, it uses the Cholesky decomposition

7.) **SOLPD**-command, $b=SOLPD(X'y,X'X)$, SOLPD uses the Cholesky decomposition

With special commands up to 16 decimal digits of the results can be displayed. All numbers in GAUSS are stored in double precision floating point format, and each takes up 8 bytes (=64 bits) of memory.

The **PRCSN**-command, *PRCSN n*, allows the computational precision of some of the GAUSS matrix operators (e. g. SOLPD, INVPD, /-operator) to be changed. The scalar n containing either 64 or 80 (=default).

---

5) Some people prefer writing $b=INV(X'X)*X'y$ instead of $b=INV(X'*X)*X'*y$, which is algebraically the same - but not numerically! These differences arise from a sophisticated programming bug in GAUSS.

If the matrix transpose operator ' is immediately followed by an operand, then it will be interpreted as * - see GAUSS-Manual (1988), page 58. Nevertheless $A*B'*C$ isn't the same as $A*B'C$. In the former case the default left to right evaluation of equal precedence operators will take place, in the second case $B'C$ will be calculated first. The reason is that the GAUSS-programmers simply forget to split up when necessary the (high) precedence of the '-operator in a (further high) matrix transpose precedence and a (lower) matrix multiply precedence.

But don't worry! The resulting numerical differences are very small and therefore they can be neglected.

6) See the preceding footnote.

When *PRCSN 80* is in effect, all temporary storage and all computations for some matrix operators are done in 80 bits. When the operator is finished, the final result is rounded to 64 bit double precision. The only exception is when intermediate results cannot be stored in 80 bits in a 64K segment. This applies to matrices larger than 80x80. In that case temporary storage is done in 64 bits instead.

2.4. 1-2-3

In 1-2-3 there are at least three ways to compute the OLS-estimator. Firstly, the REGRESSION-facility with an automatically added constant term; secondly, the REGRESSION-facility in homogeneous regression mode (the user can add a constant term to the X-matrix); and thirdly, a combination of the MATRIX-manipulation-facilities.

With special commands up to 18 decimal digits of the results can be displayed. Lotus 1-2-3 uses at least 4 up to 16 bytes of the memory to represent a number.

2.5. EXCEL

In EXCEL it is possible to compute the OLS-estimator with a combination of matrix functions (including a matrix inverse function) or with the **RGP**-function[7].

With special commands up to 15 decimal digits of the results can be displayed. Microsoft Excel stores numbers to 15 decimal digits of accuracy (the so-called full precision).

---

7) In the English versions of EXCEL this function is called LINEST. RGP is the name used in the German versions.

2.6. ASEASY

Two versions of ASEASY[8] (3.00 and 4.00b) have been available to the author. In both there are two possibilities to perform OLS-estimation, the **E – Solve**-facility (for solving the normal equations) and the matrix inverse function, both in combination with other matrix functions.

With special commands up to 11 decimal digits of the results can be displayed.

2.7. MathCAD

In MathCAD OLS-estimation can be performed either using a combination of matrix functions or using the **MINERR**-function for solving the equations $Xb = y$ by least squares in exceptional cases.

With special commands up to 16 decimal digits of the results can be displayed.

2.8. SHARP Pocket Computer PC-1403

The **MATRIX OPERATION**-facility at the SHARP Pocket Computer PC-1403 contains some basic matrix functions (e. g. invert, transpose, multiply), which can be used to calculate the OLS-estimates. The matrix invert function is based on a simple elimination procedure.

At the Sharp Pocket Computer PC-1403 up to 10 decimal digits of the results will be displayed, approximately 12 decimal digits will be processed internally.

Within the MATRIX OPERATION-facility 8 bytes are used to store a number (=matrix element).

---

8) It should be pointed out that ASEASY is common shareware.

3. Using Longley's Data Set to Assess the OLS-Accuracy of the Programs under Consideration

3.1. Permutating the Order of the X-Variables

Now Longley's data set (see table 1.1) and the OLS-routines described in chapter 2. will be used to compute OLS-estimates.

There is some reason to believe that from a numerical point of view a regression of Y onto e. g. X1, X2 isn't the same as a regression of Y onto X2, X1. In this context the position of the constant term in the X-matrix is considered to be crucial. If the constant term takes the first/last place in the X-matrix, then the numerically most stable/worst results will be expected normally.

Taking this fact into account most of the OLS-routines had to run four times, each time the order of the X-variables had been permutated.

Variable ordering variants:

a.) X1 X2 X3 X4 X5 X6 X7

b.) X2 X3 X4 X5 X6 X7 **X1**

c.) X1 **X7** X2 X3 X4 X5 X6

d.) X2 X3 X4 X5 X6 **X1** X7

For the findings of these experiments see table 3.1.

	Variable ordering variants				
	auto.	a	b	c	d
IAS-System mainf., *OLS	b	6- 8	4- 6	5- 9	3- 6
IAS-System PC, *OLS	b	5- 7	2- 5	5- 6	2- 5
RATS, LINREG	——	8-10	8-10	8-10	8-10
RATS, MATRIX	——	8-11	8-10	7-10	8-10
GAUSS, OLS	a	8-11	——	8-11	——
GAUSS, OLSQR	——	11-13	11-13	11-13	11-13
GAUSS, "/" (variant I)	——	8-10	8-10	8-10	8-10
GAUSS, "/" (variant II)	——	10-12	10-11	11-14	11-13
GAUSS, INV	——	8-11	7-10	7-11	5-10
GAUSS, INVPD	——	8-11	8-11	8-10	8-10
GAUSS, SOLPD	——	8-11	8-11	8-11	8-11
1-2-3, inhom.REGRESSION	a	9-11	——	——	——
1-2-3, homog.REGRESSION	——	9-11	10-12	9-11	10-12
1-2-3, MATRIX	——	9-11	10-12	9-11	10-12
EXCEL, matrix functions	——	6- 8			
EXCEL, RGP	a(?)	7- 9			
ASEASY 3.00, E-Solve	——	1- 3			
ASEASY 3.00, inv. func.	——	1- 3			
ASEASY 4.00b, E-Solve	——	2- 4			
ASEASY 4.00b, inv.func.	——	2- 4			
MathCAD, matrix func.	——	6-10			
MathCAD, MINERR	——	2- 5			
SHARP, MATRIX OPERATION	——	1- 4			

Table 3.1: Numerical accuracy of several OLS-programs when using Longley's data set with different variable orders. For further explanations see the following text.

In table 3.1 much information can be found concerning the numerical accuracy of the OLS-routines under consideration.

The first column titled with "auto." tells us, how the particular program places the constant term automatically. We notice, only a few routines do this at all. In this context "a" or "b" means that the constant term will be placed automatically like in variable ordering variant a or b, i. e. in the first or last column of the X-matrix, respectively. The question mark in the EXCEL, RGP-row stands for the curious fact that the RGP-function displays its results in reversed a-order.

The columns titled with "a", "b", "c" and "d" contain the range of the numerical accuracy of the particular OLS-facility according to the specific variable ordering variant. The numerical accuracy is measured as the number of identical decimal leading digits by comparing an OLS-routine-result with Longley's correct solution (see table 1.2).

For instance let the correct result be 730.192. If we compute 730.167, then we will have 4 digits accuracy. If we compute 73,019.2 or 0.073543, we will count 0 digits accuracy in both cases, because there are no identical leading digits[9].

Unfortunately, this numerical accuracy measuring method is only of an approximative nature.

For illustration let's use an extreme example. If we compare 738.735 and 729.944 with the correct 730.192, we will count 2 and 1 digits respectively.

Now we define a more precise method for numerical accuracy measurement.

---

9) To make it clear: When we compare 73,019.2 and 0.073543 with 730.192, we actually compare 73,019.2 with 00,730.192 and 000.073543 with 730.192 respectively.

Definition[10]: Given two real numbers s, cs and an integer b, where b is the decimal exponent[11] of cs in scientific notation $\pm\#.\#\#\#E\pm\#\#$. Then s is said to be numerical accurate to j° decimal digits in regard of cs, whereby j° is the biggest integer[12] j in the expression $|s-cs| \leq 5 \cdot 10^{-j+b}$.

For 738.735 and 729.944 an exact numerical accuracy of 1 and 3 digits is calculated respectively[13].

It's easy to see that the exact procedure is much more complicated than the approximative one, so we decide to use in this paper the approximative one hoping thereby for an error compensational effect in the long run.

At this time it should be clear what e. g. "7-10" in column c, row RATS, MATRIX in table 3.1 means: The MATRIX-command in RATS calculates the seven OLS-coefficients of Longley's c-ordered data set with an (approximative) numerical accuracy of at least 7 and at most 10 decimal digits respectively.

In table 3.2 the effect of the GAUSS-option *PRCSN 64* can be studied. The variables are in a-order, the entry "n. a." means, that the particular GAUSS-command is not affected by this option. (As a matter of fact the PRCSN-option only affects GAUSS-commands, which are based on the Cholesky decomposition.)

---

10) This definition has a sometimes quite desirable characteristic: It allows negative numerical accuracy.

11) If $cs=0$, then $b=0$ too.

12) In the special case $|s-cs|=0$ the value of j° should be set to the approximate number of decimal digits with which the program computes.

13) It should be pointed out, that there are even more precise accuracy-measuring-methods available. For instance Wampler (1970) has used such a still more advanced method, which results in 1.93 and 3.47 digits accuracy respectively, when applied to our small example.

GAUSS, OLS	8-11
GAUSS, OLSQR	n. a.
GAUSS, "/" (variant I)	n. a.
GAUSS, "/" (variant II)	8-11
GAUSS, INV	n. a.
GAUSS, INVPD	8-10
GAUSS, SOLPD	8-11

Table 3.2: GAUSS-option *PRCSN 64,* the variables are in a-order.

What conclusions can be drawn from tables 3.1 and 3.2?

- The hypothesis of numerically bad results when placing the constant term in the last column of the X-matrix has been only confirmed in two cases (IAS, GAUSS-INV); on the other hand 1-2-3 contradicts this theory severely.
We think a possible explanation for this fact could be the following: The constant-last-column-hypothesis seems to be only true, if the given OLS-routine uses a matrix inversion algorithm for general invertible matrices (e. g. the Gauss method or the Gauss-Jordan method or Crout's decomposition).

- There is some evidence that the programmers of the GAUSS-OLS-procedure have had knowledge of the possibly fatal effect of a disadvantageously placed constant term, although their fear seems to be unfounded in the light of the above considerations, because GAUSS-OLS is based on the Cholesky decomposition.
In case of inhomogeneous regression GAUSS-OLS always puts the constant term in the first column of the X-matrix. If the user defines a constant (which need not be equal to one) and puts it somewhere in the

X-matrix, GAUSS-OLS throws away its own constant-term and puts the user-defined one at the first place of the X-matrix.
In case of homogeneous regression the situation is quite similar. GAUSS-OLS always detects a user-defined constant term anywhere in the X-matrix and puts it at the first place.
Because of this reason the variants "b" and "d" couldn't be computed with GAUSS-OLS.

- There is a strong evidence that the *OLS-command of the IAS-System has been programmed badly. Either a matrix-inversion-algorithm of inferior quality has been used or slight programming bugs like unnecessary rounding of provisional results have been occured or most likely a combination of these and other failures are in action[14]. Otherwise a program, which runs on a mainframe computer or on a PC respectively, must not compute so unsatisfyingly.
Sarcastically speaking the PC-adaptation of the IAS-System is more honest than the mainframe-adaptation, because the former can only display up to 7 decimal digits of its results ...

- The numerical performance of RATS, GAUSS, EXCEL and MathCAD seems to be ordinary PC-software standard.

- The big positive surprise is 1-2-3! It performs quite well numerically, although OLS-estimation certainly is not its intended main purpose.
It should be noted that the results of each of the three OLS-computing-variants of 1-2-3 are absolutely equivalent up to the last displayed digit after the comma. So in this paper any further 1-2-3 calculations will be done with the more effortless REGRESSION-facility only.

- The old golden rule of thumb has been confirmed again through ASEASY: *Beware of shareware!!* Strictly speaking it's incredible: A

14) The fact of the automatically wrong placed constant term is rather a consequence of a failure (inferior matrix-inversion-algorithm) than a failure in itself. Basing the *OLS-command on e. g. Cholesky or QR decomposition should eliminate this unfortunate effect, provided that our foregoing speculation about the constant-last-column-hypothesis is correct.

program for a 386-PC has the same numerical accuracy as a pocket calculator with 6K memory.

- We have been quite surprised that the PRCSN-option only slightly affects the GAUSS-results[15]. It seems that the differences due to less precision are in most cases only one hundredth or one thousandth of the differences due to the matrix-inversion-method (Cholesky decomposition).

To summarize: In order to compute numerically accurate OLS-estimates the GAUSS-procedures OLSQR and "/" (var. II) can be recommended. When using "/" (var. II) be sure that *PRCSN 80* is in action.

Also the OLS-results of the LOTUS spread-sheet 1-2-3 seem to be o. k. from a numerical point of view.

---

15) One curious fact should be reported: The "/"-operator (var. II) and the SOLPD-command perform quite different, when *PRCSN 80* is in action. But with *PRCSN 64* both routines compute the same results up to 16 digits displayable.

3.2. Transforming the Variables

In the next experiments Longley's variables `X2, ... X7, Y` will be transformed linearly.

Variable transforming variants[16]:

e.) the variable minus its first observation
f.) the variable minus its midrange
g.) the variable minus its mean, inhomogeneous regression
h.) the variable minus its mean, homogeneous regression

All these linear transformations affect only the value of the intercept. The true, i. e. untransformed intercept `b1` can be calculated with the following formula:

$$b1 = b1t + cy - c2 \cdot b2 - \ldots - c7 \cdot b7 \qquad (3.1)$$

`b1t` ... the intercept[17] of the OLS-regression with transformed variables
`cy` .... the value which is subtracted from `Y`
`c2` .... the value which is subtracted from `X2`
...
`c7` .... the value which is subtracted from `X7`

Consequently the OLS-data matrix [X|y] has the form $[X1,\ X2-c2 \cdot 1_{16},\ \ldots,\ X7-c7 \cdot 1_{16} | Y-cy \cdot 1_{16}]$ in variants "e", "f" and "g", and it has the form $[X2-c2 \cdot 1_{16},\ \ldots,\ X7-c7 \cdot 1_{16} | Y-cy \cdot 1_{16}]$ in variant "h", $1_{16}=(1,\ \ldots,\ 1)'$.

For the findings of these latest experiments see table 3.3.

---

16) Let's motivate these transformations with an example. The variable `X7` ranges from 1947 to 1962, but nothing will be changed substantially, if it ranges by way of example from 47 to 62 or from 0 to 15. We recognize, the original `X7` contains a lot of garbage, which subsequently lead to unnecessary big OLS-provisional results, which probably can't be stored with full accuracy because of limited space. We see, such transformations help to eliminate avoidable failures.

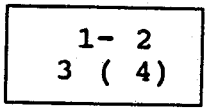
17) `b1t` equals zero at variant "h". At variant "g" it should equal zero too, but in computed reality it is only a number *near* to zero.

	Variable transforming variants			
	e	f	g	h
IAS-System mainframe, *OLS	6- 8 7	6- 8 7	6- 8 7 (6)	6- 8 7
IAS-System PC, *OLS	5- 7 6	5- 7 6	5- 7 6 (5)	5- 7 6
RATS LINREG	11-14 13	11-14 13	12-15 13 (14)	12-15 13
RATS MATRIX	11-13 12	12-14 13	12-14 14 (13)	12-14 14
GAUSS OLS	11-13 13	12-15 13	12-15 13 (14)	12-15 13
GAUSS OLSQR	13-15 15	12-15 14	13-15 15 (14)	13-16 15
GAUSS "/" (variant I)	12-13 13	12-14 13	12-14 14 (14)	12-14 14
GAUSS "/" (variant II)	14-16 16	14-16 15	14-16 15 (14)	14-16 15
GAUSS INV	11-14 13	12-13 12	13-14 13 (14)	13-14 13
GAUSS INVPD	11-14 13	12-14 13	12-15 13 (14)	12-15 13
GAUSS SOLPD	12-14 14	12-14 14	12-15 14 (14)	12-15 14
1-2-3 REGRESSION	14-16 15	14-16 16	14-17 16 (17)	14-17 16

Table 3.3: Numerical accuracy of several OLS-programs when using different linear transformations of Longley's data set. For further explanations see the following text.

In table 3.3 a lot of information can be found about the (approximative) numerical accuracy of diverse OLS-routines when applied to transformed versions of Longley's data set.

Each cell of the table contains either two or three entries. Let's explain their meaning through an example:



In this example the OLS-coefficients `b2 ... b7` have a numerical accuracy of 1 - 2 decimal digits. The OLS-coefficient `b1` (see formula 3.1) has a numerical accuracy of 3 decimal digits. The figure 4 (which is putted in parentheses) makes only sense at variant "g". It means, that the OLS-coefficient `b1t` (which has a true value of zero at variant "g") contains 4 leading zeros, hence the number looks like 0.000xxx.

What conclusions can be drawn from table 3.3?

- Rather simple transformations can substantially improve the numerical accuracy of an OLS-routine.

- It seems, that mean-centering of the variables is slightly better than the variants "e" and "f". It is not really obvious, whether "g" or "h" should be prefered. No doubt, "h" leads to a smaller matrix to invert, but on the other side the constant term in "g" appears to bring in some (more or less necessary) flexibility.

- Again, 1-2-3 is the big positive surprise.

- The numerical performance of the IAS-System still remains inferior. A closer look shows that in both adaptations (mainframe computer and PC) the outcomes of variants "a" and "g" are exactly the same with the exception of the coefficient `b1t`, which doesn't appear in "a" of course.

Obviously mean-centering of the variables is usual within the *OLS-command of the IAS-System.
But this new finding makes our assessment of the numerical performance of the *OLS-command from bad to worse, because the entries in the IAS-System-rows of table 3.1 are no longer relevant. The differences in numerical accuracy of the IAS-System to RATS, GAUSS and 1-2-3 must now be learned from table 3.3, strictly speaking from the columns "g" and "h" of this table. Because only at these two variants the automatic mean-centering of the *OLS-command can be considered to have a negligible numerical effect.

Let's summarize: After throwing away the numerical garbage of the variables the numerical performance of RATS, GAUSS and 1-2-3 becomes quite acceptable.

On the other hand the IAS-System should be used cautiously.

4. Condition Numbers

4.1. Motivation

In the standard linear model

$$y = X\beta + \epsilon \qquad E(\epsilon) = 0 \qquad E(\epsilon\epsilon') = \sigma^2 I_n \tag{4.1}$$

where y, X, β and ϵ are nx1, nxp, px1 and nx1 matrices respectively, the least squares estimator of β is given by

$$b = (X'X)^{-1}X'y. \tag{4.2}$$

If small changes are made to the values taken by the dependent variable $y \rightarrow y + \Delta y$, then the least squares estimator will also be subject to small changes $b \rightarrow b + \Delta b$, where

$$\Delta b = (X'X)^{-1}X'(\Delta y). \tag{4.3}$$

The proportionate effect of these changes may be measured by the general formula[18]

$$\frac{\|\Delta b\| \ / \ \|b\|}{\|\Delta y\| \ / \ \|y\|} \tag{4.4}$$

where the choice of convenient norms is at the disposal of the investigator. In particular if the L_2-norm is adopted in all four cases, then this expression becomes

$$\frac{[(\Delta b_1)^2 + \ldots + (\Delta b_p)^2]^{1/2} \ / \ [b_1^2 + \ldots + b_p^2]^{1/2}}{[(\Delta y_1)^2 + \ldots + (\Delta y_n)^2]^{1/2} \ / \ [y_1^2 + \ldots + y_n^2]^{1/2}}. \tag{4.5}$$

---

18) However, it should be noted that the practitioner is not restricted to formula (4.4), although formula (4.4) does give the elasticity of b, with respect to y, when $n = 1$ and $p = 1$.

If expression (4.5) is maximized over Δy (for given y), then *Farebrother's single limit condition number*[19] (Farebrother's SLCN) will be obtained.

If expression (4.5) is maximized over b and Δy, when $y=X\beta$ (and not just over Δy for given y), then it will give rise to the *double limit condition number* (DLCN), also called the *traditional condition number*[20].

When using e. g. Longley's data set for statistical data analysis we prefer to adopt a condition number which better reflects the situation with which we are concerned. Now the value of Δy is completely unknown, but the value of y is known, at least approximately, and we therefore prefer to adopt Farebrother's SLCN, which treats the value of y as if it were known exactly, rather than the double limit condition number. In particular, we note that the observed value of y does not feature in the calculations for the double limit condition number.

4.2. Farebrother's Empirical Single Limit Condition Number and Derived Summary Statistics

In order to find the theoretical values of Farebrother's SLCN and the DLCN for a given data set, some results of the theory of eigenvalues had to be applied.

The corresponding empirical condition numbers do not make use of eigenvalues as the necessary maximizations are performed empirically[21] rather than formally. Thus Farebrother's empirical SLCN is the largest value of expression (4.5) that the user is able to find for a range of values of Δy.

---

19) See Farebrother (1988), p. 168.

20) See Farebrother (1988), p. 169.

21) These empirical condition numbers will tend to be smaller than the corresponding theoretical condition numbers as the latter attain the formal maximums. However, they also ignore the impact of the hardware/software-combination on the calculations, so it is possible for empirical condition numbers to be larger than the corresponding theoretical condition numbers.

The following four experiments (which are based on Longley's data set) are distinct from the ideas of influential data analysis, which is concerned with the effect of one or more very large disturbances, whereas we are concerned with the effect of a large number of very small disturbances. Our experiments can of course be generalized to small variations in the full [Xy] data matrix, but there is, at present, no generally accepted measure for the effects of such changes.

In the following all the least squares computations are done with the GAUSS-procedure OLSQR. At the beginning of each experiment the seed for the GAUSS-random-number-generator has been given the value 79319.

Experiment 1n: The variables in the X-matrix have been "a"-ordered (see chapter 3.1.), and the Y-variable has been disturbed with a vector of N(0,0.5) distributed random numbers. The disturbance of Y has been repeated 10,000 times, and each time expression (4.5) has been calculated. In table 4.1 a summary statistic of this computations can be found.

Mean	Std.Dev.	Variance	Minimum	Maximum	#Obs.
44.3711	32.1343	1032.6142	0.0066	194.2920	10000

Table 4.1: A summary statistic of experiment 1n

Experiment 1u: The only difference to experiment 1n is the usage of U(-0.5,0.5) uniformly distributed random numbers instead of N(0,0.5) distributed ones. See table 4.2 for the findings.

Mean	Std.Dev.	Variance	Minimum	Maximum	#Obs.
45.0943	32.0507	1027.2461	0.0271	170.7619	10000

Table 4.2: A summary statistic of experiment 1u

By comparing the experiments 1n and 1u we find 194.3 as Farebrother's empirical SLCN for Longley's data set. This is quite a satisfying result, because Farebrother's theoretical SLCN for this particular problem is 219.4, which can be empirically obtained too, when using the vector[22]

```
-1.0537211494320050E-02
 3.0456278979339730E-02
-3.7279161517160800E-02
-7.3829249807598990E-03
 6.2922919813453610E-01
 2.4679411247153260E-01
-3.0646087415779010E-01
-1.9215282624859270E-01
-4.9310282956503650E-02
-3.8400734682145700E-01
-3.7081429524225880E-01
-2.5574286825047540E-02
 1.7260157104549300E-01
-9.4795017108799950E-02
 1.5990445073726200E-01
 2.3967098684884080E-01
```

to disturb the Y-variable. We see, this specific vector only consists of small numbers - our chosen strategy (only to use small variations when calculating Farebrother's empirical SLCN) seems to be reasonable in Longley's case.

It should be noted that the order of the components of this particular vector is crucial, because if this order is severely disarranged through permutation, then quite normal outcomes can be expected. E. g. putting away the first component from it's place and appending it at the end results in a value of 68.67 for expression (4.5), or reversing the whole vector results in a value of 10.60, and so forth.

---

22) This is the eigenvector corresponding to the largest eigenvalue of the matrix $X(X'X)^{-2}X'$. All calculations of eigenvalues and -vectors are done with the GAUSS-procedure EIGRS2.

Experiment 2n: The variables in the X-matrix have been in "h"-order (see chapter 3.2.), and the Y-variable has been disturbed with a vector of N(0,0.5) distributed random numbers. The disturbance of Y has been repeated 10,000 times, and each time expression (4.5) has been calculated[23]. In table 4.3 a summary statistic of this computations can be found.

Mean	Std.Dev.	Variance	Minimum	Maximum	#Obs.
2.4272	1.6158	2.6108	0.0128	9.9711	10000

Table 4.3: A summary statistic of experiment 2n

Experiment 2u: The only difference to experiment 2n is the usage of U(-0.5,0.5) distributed random numbers instead of N(0,0.5) distributed ones. See table 4.4 for the findings.

Mean	Std.Dev.	Variance	Minimum	Maximum	#Obs.
2.4581	1.6137	2.6039	0.0127	8.7577	10000

Table 4.4: A summary statistic of experiment 2u

Again, the empirical single limit condition number 9.97 is quite near to the theoretical value of 11.1, which can empirically be obtained through the usage of the vector[22]

```
-7.5802380136535300E-03
 3.5456340462030870E-02
-3.6203535439509810E-02
-1.3845231308676900E-02
 6.2654903623474530E-01
```

23) It should be pointed out that $p=6$ now.

```
 2.4754413562061830E-01
-2.9947824555949090E-01
-1.8930513195025260E-01
-5.8931780443459480E-02
-3.8939282766476990E-01
-3.7063514272743530E-01
-2.4987069218821820E-02
 1.6458798970895870E-01
-9.6235691232370180E-02
 1.6553472263994440E-01
 2.4692266889214630E-01
```

to disturb the Y-variable. Once again we see, how adequate our chosen many-small-disturbances-strategy for Longley's data set has been, this vector only consists of small values too. It would be interesting to know, how our strategy works in the general case.

Let's summarize: In principle the outcomes of these experiments confirm those of the third chapter. Throwing away the numerical garbage through a method, which only affects the mostly meaningless value of the intercept, causes more stable results - just indicated through smaller values of Farebrother's (theoretical and empirical) SLCN.

Remark: The traditional condition number (DLCN) for a range of data matrices has been tabulated by Späth (1987), p. 44. For Longley's data set the values $4.859 \cdot 10^9$ and $5.769 \cdot 10^5$ have been specified for the variable ordering/transforming variants "a" and "h", respectively.

5. Using Longley's X-Variables to Assess the Effect of Repeated Matrix Inversions

5.1. The X-Variables Are Untransformed

Only a few statistical problems have an one-step-solution like the OLS-regression-problem. For many questions numerical interpolation is necessary to get a solution.

Having this in mind we look at the preceding results and worry about the numerical accuracy of any interpolation result. Since, if an interpolation algorithm stops, does it really stop, because it has approximately reached the desired minimum/maximum/..., or because it has walked into a numerical trap?

In the light of these considerations we have planned the following experiment:

i) $A=X'X$

ii) $B=(A^{-1})^{-1}$, $A=B$

iii) Repeat step ii) many times and see what happens[24].

See tables 5.1 and 5.2 for the first results when using GAUSS, INVPD and GAUSS, INV for matrix inversion and Longley's data set to form the X-matrix, X=[X1, ..., X7].

---

24) This experiment isn't really a direct answer to the above-mentioned considerations about the numerical accuracy of interpolation results.

Because no serious algorithm inverts the same matrix twice, but very often the following is done: An inverted matrix is used to form a new matrix, which after its inversion is used to form a new matrix, which after its inversion is used to form another new matrix, and so on.

So we think, it would be interesting to see, if there are systematic distortions or drifts caused through the inversion-procedure in use. In case such distortions can be detected in our artificial case, then this problem could be of some significance for the more general cases, too.

Iter-ation	Sum of deviat. from the previous	Difference between sum of abs. deviations & abs. sum of deviat. from the previous	Sum of deviat. from the origin	Difference between sum of abs. deviations & abs. sum of deviat. from the origin
1	-23863.8	0.000e+00	-23863.8	0.000e+00
2	108575.1	0.000e+00	84711.3	0.000e+00
3	-35562.9	0.000e+00	49148.3	0.000e+00
4	-54599.8	0.000e+00	-5451.5	0.000e+00
5	-47432.1	0.000e+00	-52883.5	0.000e+00
6	-71387.0	0.000e+00	-124270.5	0.000e+00
7	-101539.8	0.000e+00	-225810.2	0.000e+00
8	10317.0	0.000e+00	-215493.2	0.000e+00
9	10976.8	0.000e+00	-204516.4	0.000e+00
10	12841.3	0.000e+00	-191675.1	0.000e+00
11	79158.7	0.000e+00	-112516.5	0.000e+00
12	161852.5	0.000e+00	49336.1	0.000e+00
13	67440.6	0.000e+00	116776.7	0.000e+00
14	63942.3	0.000e+00	180719.0	0.000e+00
15	80641.4	0.000e+00	261360.4	0.000e+00
16	-11596.7	0.000e+00	249763.7	0.000e+00
17	24062.0	0.000e+00	273825.7	0.000e+00
18	-20328.3	0.000e+00	253497.4	0.000e+00
19	39983.9	0.000e+00	293481.3	0.000e+00
20	47168.6	0.000e+00	340649.9	0.000e+00

Table 5.1: Inversion and Reinversion of X'X with GAUSS, INV

Iter-ation	Sum of deviat. from the previous	Difference between sum of abs. deviations & abs. sum of deviat. from the previous	Sum of deviat. from the origin	Difference between sum of abs. deviations & abs. sum of deviat. from the origin
1	-65133.2	0.000e+00	-65133.2	0.000e+00
2	-90455.2	0.000e+00	-155588.3	0.000e+00
3	-42104.2	0.000e+00	-197692.6	0.000e+00
4	11281.8	0.000e+00	-186410.8	0.000e+00
5	-11555.0	0.000e+00	-197965.8	0.000e+00

6	102715.3	0.000e+00	-95250.4	0.000e+00
7	-27779.0	0.000e+00	-123029.4	0.000e+00
8	-30207.6	0.000e+00	-153237.1	0.000e+00
9	43535.6	0.000e+00	-109701.5	0.000e+00
10	99380.9	0.000e+00	-10320.6	0.000e+00
11	63195.7	0.000e+00	52875.1	0.000e+00
12	-40860.8	0.000e+00	12014.4	0.000e+00
13	55608.9	0.000e+00	67623.3	0.000e+00
14	39417.7	0.000e+00	107041.0	0.000e+00
15	-63416.0	0.000e+00	43624.9	0.000e+00
16	104915.1	0.000e+00	148540.0	0.000e+00
17	-21727.4	0.000e+00	126812.6	0.000e+00
18	60607.3	0.000e+00	187419.9	0.000e+00
19	105334.6	0.000e+00	292754.5	0.000e+00
20	-76800.9	0.000e+00	215953.6	0.000e+00

Table 5.2: Inversion and Reinversion of X'X with GAUSS, INVPD

The sum of deviations of a matrix M1 from a matrix M0 is defined as $\Sigma_i\Sigma_j(M1_{ij}-M0_{ij})$. The sum of absolute deviations is $\Sigma_i\Sigma_j|M1_{ij}-M0_{ij}|$. If the difference between the sum of absolute deviations and the absolute sum of deviations is equal to zero, it follows that either $M1_{ij}\geq M0_{ij}$ or $M1_{ij}\leq M0_{ij}$, for all i, j.

Next, we have enlarged this experiments (100,000 iterations), see tables 5.3, 5.4 and 5.5 for the findings.

	GAUSS, INV		GAUSS, INVPD	
Iter-ation	Sum of deviat. from the previous	Sum of deviat. from the origin	Sum of deviat. from the previous	Sum of deviat. from the origin
5000	100708.4	2110803.2	-76521.7	-7725027.2
10000	-71628.4	3599665.8	53987.4	-9573010.8
15000	-21985.7	-3674926.3	101175.2	-8256405.9
20000	62399.2	1936206.7	-13105.5	-8784136.3
25000	82937.3	10425216.6	55158.6	-8534777.4
30000	-13689.8	12093280.2	-36881.6	3253689.7
35000	-138136.0	9686586.6	-32750.4	-312355.1
40000	61272.4	4727668.7	56374.2	6913165.5
45000	-61111.3	5381395.8	36637.1	6892289.6
50000	12099.7	5673383.3	86231.6	-8111959.9

55000	-14291.0	6106171.8	-63863.9	-1051851.3
60000	51356.2	2557772.1	-85446.1	-6588042.7
65000	41314.4	-2183181.3	-78978.3	-2771173.7
70000	-50226.4	743829.6	-21166.0	-2456247.9
75000	-48185.6	-740084.8	33137.2	-8562268.1
80000	-30936.8	1774699.6	74099.1	2464447.2
85000	-34857.9	2797461.6	-201023.0	236917.8
90000	38479.0	141801.2	146955.0	-307384.0
95000	-48566.0	1149481.4	-158613.2	4538168.4
100000	6848.0	5446346.4	-127423.4	1094966.1

Table 5.3

	GAUSS-proc.	Attained at iter.	Value
Largest sum of absolute deviations from the previous	INV	71514	178269.45
	INVPD	90444	234457.40
Smallest sum of absolute deviations from the previous	INV	51010	2.24
	INVPD	86123	1.23
Largest Difference between sum of abs. deviations and abs. sum of deviat. from the previous	INV	23782	21.08
	INVPD	77642	13.93
Largest sum of absolute deviations from the origin	INV	31015	14401074.51
	INVPD	18189	13758062.45
Smallest sum of absolute deviations from the origin	INV	2638	209.71
	INVPD	57192	1456.84
Largest Difference between sum of abs. deviations and abs. sum of deviat. from the origin	INV	90014	2425.46
	INVPD	67454	4512.23

Table 5.4

	GAUSS-procedure	Number
Number of Differences not equal to zero between sum of abs. deviat. & abs. sum of deviat. from the prev.	INV	27
	INVPD	49
Number of Differences not equal to zero between sum of abs. deviat. & abs. sum of deviat. from the orig.	INV	34
	INVPD	85

Table 5.5

The interpretations of the results of tables 5.1 to 5.5 will be made in chapter 5.2..

5.2. The X-Variables Are Mean-Centered

Now we have formed the X-matrix from Longley's mean-centered variables X2 ... X7. Let's look at the findings.

Iteration	Sum of deviat. from the previous	Difference between sum of abs. deviations & abs. sum of deviat. from the previous	Sum of deviat. from the origin	Difference between sum of abs. deviations & abs. sum of deviat. from the origin
1	1.704e-02	0.000e+00	1.704e-02	0.000e+00
2	1.199e-02	0.000e+00	2.903e-02	0.000e+00
3	-1.236e-02	0.000e+00	1.667e-02	0.000e+00
4	-2.004e-04	-8.389e-06	1.647e-02	0.000e+00
5	-7.087e-04	-2.660e-06	1.576e-02	-2.581e-05
6	-8.940e-03	0.000e+00	6.823e-03	-3.252e-05
7	-6.950e-03	0.000e+00	-1.276e-04	-3.330e-06
8	-1.828e-02	0.000e+00	-1.841e-02	0.000e+00
9	-8.597e-04	-3.386e-05	-1.927e-02	0.000e+00
10	-1.371e-02	0.000e+00	-3.298e-02	0.000e+00

11	2.235e-03	0.000e+00	-3.075e-02	0.000e+00
12	-1.457e-02	0.000e+00	-4.532e-02	0.000e+00
13	1.795e-02	0.000e+00	-2.737e-02	0.000e+00
14	-1.373e-02	0.000e+00	-4.110e-02	0.000e+00
15	4.100e-03	-8.508e-05	-3.700e-02	0.000e+00
16	9.390e-03	0.000e+00	-2.761e-02	0.000e+00
17	1.164e-02	0.000e+00	-1.597e-02	0.000e+00
18	5.664e-03	0.000e+00	-1.031e-02	0.000e+00
19	-4.221e-03	-2.998e-06	-1.453e-02	0.000e+00
20	6.781e-03	0.000e+00	-7.746e-03	0.000e+00

Table 5.6

Iteration	Sum of deviat. from the previous	Difference between sum of abs. deviations & abs. sum of deviat. from the previous	Sum of deviat. from the origin	Difference between sum of abs. deviations & abs. sum of deviat. from the origin
1	4.332e-02	0.000e+00	4.332e-02	0.000e+00
2	1.399e-02	0.000e+00	5.731e-02	0.000e+00
3	-2.904e-02	0.000e+00	2.827e-02	0.000e+00
4	8.112e-03	0.000e+00	3.638e-02	0.000e+00
5	-4.722e-02	0.000e+00	-1.084e-02	0.000e+00
6	-1.973e-02	0.000e+00	-3.057e-02	0.000e+00
7	2.410e-02	0.000e+00	-6.470e-03	0.000e+00
8	5.559e-02	0.000e+00	4.912e-02	0.000e+00
9	2.601e-02	0.000e+00	7.513e-02	0.000e+00
10	-3.577e-02	0.000e+00	3.936e-02	0.000e+00
11	-5.073e-02	0.000e+00	-1.137e-02	0.000e+00
12	-2.044e-02	0.000e+00	-3.181e-02	0.000e+00
13	7.145e-02	0.000e+00	3.964e-02	0.000e+00
14	1.669e-02	0.000e+00	5.633e-02	0.000e+00
15	-2.810e-02	0.000e+00	2.822e-02	0.000e+00
16	5.987e-02	0.000e+00	8.810e-02	0.000e+00
17	1.402e-02	0.000e+00	1.021e-01	0.000e+00
18	4.198e-02	0.000e+00	1.441e-01	0.000e+00
19	3.918e-03	-2.086e-07	1.480e-01	0.000e+00
20	1.873e-02	0.000e+00	1.668e-01	0.000e+00

Table 5.7

Iter-ation	GAUSS, INV		GAUSS, INVPD	
	Sum of deviat. from the previous	Sum of deviat. from the origin	Sum of deviat. from the previous	Sum of deviat. from the origin
5000	0.010	1.288	-0.001	-2.784
10000	0.009	0.738	0.040	-2.652
15000	0.010	-0.652	0.031	-3.952
20000	0.015	0.303	0.066	-2.467
25000	0.003	0.984	0.028	-1.271
30000	-0.003	2.360	0.014	-2.736
35000	0.003	2.745	0.026	-2.562
40000	-0.009	2.599	0.025	-1.286
45000	0.003	3.384	0.024	-3.952
50000	-0.015	3.939	0.001	-3.438
55000	0.005	3.530	0.021	0.522
60000	0.007	4.426	-0.003	-0.757
65000	0.003	4.021	0.015	-2.162
70000	-0.015	4.689	0.005	1.803
75000	-0.008	4.007	0.048	-2.337
80000	0.007	2.086	0.017	0.173
85000	0.021	2.636	-0.011	0.847
90000	0.004	2.796	0.036	0.665
95000	-0.014	1.194	-0.008	-0.589
100000	-0.012	-0.518	-0.036	-0.057

Table 5.8

	GAUSS-proc.	Attained at iter.	Value
Largest sum of absolute deviations from the previous	INV	22610	0.04416234
	INVPD	92513	0.09905036
Smallest sum of absolute deviations from the previous	INV	83098	0.00000580
	INVPD	96847	0.00000359
Largest Difference between sum of abs. deviations and abs. sum of deviat. from the previous	INV	95745	0.00015501
	INVPD	324	0.00025769

Largest sum of absolute deviations from the origin	INV	69141	5.37212123
	INVPD	15776	5.19677293
Smallest sum of absolute deviations from the origin	INV	7	0.00013089
	INVPD	75	0.00079712
Largest Difference between sum of abs. deviations and abs. sum of deviat. from the origin	INV	98337	0.02987833
	INVPD	97135	0.02599336

Table 5.9

	GAUSS-procedure	Num-ber
Number of Differences not equal to zero between sum of abs. deviat. & abs. sum of deviat. from the prev.	INV	19195
	INVPD	11456
Number of Differences not equal to zero between sum of abs. deviat. & abs. sum of deviat. from the orig.	INV	26802
	INVPD	30206

Table 5.10

One main message of the experiments of chapter 5.1. and 5.2. is evident. Once again, either we throw away numerical garbage or the squaring of the variables (when forming X'X) will seriously undermine the stability of the outcomes: From chapter 5.1. to 5.2. the change in the orders of magnitude of the variables amounts to 10^2 or 10^3, while the corresponding change in the orders of magnitude of the results is considerably higher.

But it's interesting to see, that the results of chapter 5.1. also show some kind of stability. E. g. it seems, that the sum of deviations from the origin always remains inside a specific range[25], i. e. there is no real evidence of a suspected systematic drift in a specific direction.

It's also worth to note, that almost all of the deviations from the previous have the same sign at chapter 5.1., while the corresponding results of chapter 5.2. are not so one-sided. We think, this last-mentioned fact is worth to attain some further investigation.

25) Should this fact be interpreted as an indication of chaos?

6. Concluding Remarks

Our main results are the following:

- There are quite large differences in the numerical accuracy of various OLS computer programs[26].

- Throwing away numerical garbage can improve the numerical accuracy of the OLS-computations.

These statements aren't new, but people often forget it. They worry only about how fast their programs calculate, but they don't worry about how accurate these calculations are.

All the calculations, which have been done here, are based on a single data set. This is only a reasonable approach, if the magnifier-assumption (see the ending of chapter 1.) holds.

Therefore it will be of great interest to repeat the computations - especially those of chapter 5. - with other X-matrices. The author will be glad to be informed about the findings of such or similar experiments.

It will also be of great interest to examine the numerical accuracy of other procedures of several computer software packages - in particular such procedures, which use iteration-based algorithms to get their solutions.

Finally, we think that studies like this one can give support to the ordinary software user, which software package should be able to meet his conceptions of numerically accurate outcomes.

---

26) It is worth to note that each of the OLS-routines under our consideration has a numerical accuracy of at least one decimal digit when using Longley's data set. Longley (1967) has reported on routines, which perform much poorer numerically.

7. References

7.1. General References

BEATON, Albert E., RUBIN, Donald B. and BARONE, John L. (1976): The Acceptability of Regression Solutions: Another Look at Computational Accuracy. J. Am. Stat. Assoc. 71, pp. 158-168.

DENT, Warren T. and CAVANDER, David C. (1977): More on Computational Accuracy in Regression. With a comment of BEATON et al. and a rejoinder of the authors, J. Am. Stat. Assoc. 72, pp. 598-602.

ESPASA, Antoni (1977): A Note on the Acceptability of Regression Solutions: Another Look at Computational Accuracy. J. Am. Stat. Assoc. 72, p. 603.

FAREBROTHER, R. W. (1988): Linear least squares computations. Statistics: textbooks and monographs, volume 91, Marcel Dekker, Inc..

FAREBROTHER, R. W. (1991): Relative Local Influence and the Condition Number. To appear.

LONGLEY, J. W. (1967): An Appraisal of Least Squares Programs for the Electronic Computer from the Point of View of the User. J. Am. Stat. Assoc. 62, pp. 819-841.

SPÄTH, H. (1987): Mathematische Software zur linearen Regression. R. Oldenbourg Verlag.

WAMPLER, Roy H. (1970): A Report on the Accuracy of Some Widely Used Least Squares Computer Programs. J. Am. Stat. Assoc. 65, pp. 549-565.

7.2. Computer Program Manuals

EXCEL (1987): Microsoft Excel Reference. Complete Spreadsheet with Business Graphics and Database, Version 2.0. For IBM Personal System/2, IBM PC AT, and Compatibles, Microsoft Corporation.

EXCEL (1988): Microsoft Excel zum Nachschlagen. Vollständiges Tabellenkalkulationsprogramm mit Grafik und Datenbank. Für IBM Personal System/2, IBM PC/AT und Kompatible, Microsoft Corporation.

GAUSS (1988): The GAUSS System, Version 2.0, Aptech Systems, Inc..

IAS-System (1990): User Reference Manual of the IAS-System (Inter-active Simulation System), Level IAS-3.8. The IAS-System has been developed at the Institute for Advanced Studies in Vienna, Austria.

LOTUS (1989): 1-2-3 Version 3. Kurzreferenz, Lotus Development Corporation.

LOTUS (1989): 1-2-3 Version 3. Referenzhandbuch, Lotus Development Corporation.

MathCAD (1989): User's Guide of MathCAD, Mathsoft, Inc..

RATS (1989): USER'S MANUAL of RATS, Version 3.02 by Thomas A. Doan, VAR Econometrics, Inc..

SHARP (1986): Bedienungsanleitung für den Taschencomputer Modell PC-1403.