A MODIFIED AGGREGATION PROGRAM FOR
THE PILOT PROCESS INTEGRATED MODEL

by

Haruko Hirose

TECHNICAL REPORT SOL-80-31

December 1980

Research and reproduction of this report were supported by the Department of Energy Contract DE-AC03-76SF00326, PA# DE-AT03-79E110601, DE-AM03-76SF00326, PA# DE-AT03-80E110682; Electric Power Research Institute Contract RP 652:1; Institute for Energy Studies at Stanford University.

The views expressed in this document are those of the authors and NOT necessarily those of any of the sponsors.

Reproduction in whole or in part is permitted for any purposes of the United States Government. This document has been approved for public release and sale; its distribution is unlimited.
Abstract

The PILOT Process Integrated Model can produce energy/economic scenarios for time periods of up to 100 years by aggregating several 5 year time periods into one. This report presents modification to an existing aggregation method that utilize the special structure of the Consumers Energy Service Model (CESM) and the Industrial Energy Service Model (IESM) and reduce aggregation bias in these portions of the PPIM.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. Introduction</td>
<td>1</td>
</tr>
<tr>
<td>II. Modification of the Variable Time Model</td>
<td>3</td>
</tr>
<tr>
<td>A. CESM and IESM models</td>
<td>3</td>
</tr>
<tr>
<td>B. Geometric aggregation scheme</td>
<td>4</td>
</tr>
<tr>
<td>C. Modification of the variable time model program</td>
<td>5</td>
</tr>
<tr>
<td>III. Test Run of Modified Variable Time Model</td>
<td>8</td>
</tr>
<tr>
<td>A. Test run</td>
<td>8</td>
</tr>
<tr>
<td>B. Comparison of results</td>
<td>8</td>
</tr>
<tr>
<td>C. Conclusion</td>
<td>13</td>
</tr>
</tbody>
</table>
TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. GNP and Total Primary Energy Consumption Comparison</td>
<td>10</td>
</tr>
<tr>
<td>2. Space Heat Comparison</td>
<td>11</td>
</tr>
<tr>
<td>3. Other Thermal Comparison</td>
<td>12</td>
</tr>
</tbody>
</table>
ILLUSTRATIONS

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Row aggregation in the MAIN program</td>
<td>6</td>
</tr>
<tr>
<td>2.</td>
<td>Column aggregation in the subroutine UPDATE</td>
<td>7</td>
</tr>
</tbody>
</table>
I. INTRODUCTION

The Stanford PILOT Energy/Economic model provides projections of energy production and use and of economic growth in the U.S. over a 40 year span, 1973-2012, divided into eight 5 year periods. A longer time horizon of say, 100 years, would enable the PILOT model to address policy decisions whose effects may not be felt till well past the turn of the century. The decisions surrounding plutonium recycling and the fast breeder reactor fall in such a category, and have been studied using a longer time horizon by Avi-Itzhak and Connolly [1]. A longer time horizon for the PILOT model is also useful for determining terminal capital stocks and other end conditions for the shorter 40 year time horizon.

However, it is not practical computationally to run a scenario of 20 periods of 5 years each. To overcome this difficulty, a computer program has been developed and tested to aggregate the 20 time periods into a smaller number of planning periods of variable length, yielding a LP matrix the size of the 40-year PILOT model [2]. The length of any time period in the aggregated matrix is some multiple of 5 years.

The aggregation scheme consists of two steps.

- Aggregating variables (by adding column coefficients).
- Aggregating equations (by adding row coefficients).

It has been shown that the solution yielded by the reduced problem is consistent with that of the original problem but not necessarily conversely [3].

The aggregation scheme substitutes one planning period for several periods of the original matrix. The activity levels in the aggregated
periods are intended to be representative of similar activities in the several unaggregated periods.

Since the date this scheme was first implemented, some modifications to the PILOT model have been made. A Consumers Energy Service Model (CESM) [4] and an Industrial Energy Service Model (IESM) [5] have been added to the PILOT model, forming the PILOT Process Integrated Model. These two submodels utilize energy facility capital stock accounting different from that in the main model. This capital stock modeling leads to LP columns with exponentially declining coefficients in later periods, and suggests that a somewhat different aggregation scheme may help decrease aggregation bias in the CESM and IESM portion of the integrated model. The CESM and IESM together contain approximately 320 rows and 1000 columns of the total 1300 rows and 2700 columns in an eight-period PILOT matrix. An aggregation scheme that can reduce bias in this fraction of the total model should yield improved results for the whole as well.
II. MODIFICATION OF THE VARIABLE TIME MODEL

A. CESM and IESM models

Many CESM and IESM variables refer to the total amount of capacities installed in the current period. Fractions of these capacities survive to be used in latter periods. The capacities depreciate according to an exponential curve, for example, if the coefficient of a column in period $t$ is 1, the coefficient in period $t + k$ is $d^k$ where $0 < d < 1$ is the survival fraction from one 5-year period to the next.

Consider the example of an energy technology $T$ installed in period 1. Suppose periods 2, 3 and 4 are aggregated to a single 15 year period. Since midpoints of the planning periods are used as representative dates, the contribution of technology $T$ in the aggregated period should be given by a survival fraction based on 2 full time periods, or $d^2$. A scheme of choosing the coefficient according to an arithmetic average would give a coefficient equal to $\frac{d + d^2 + d^3}{3}$. If the geometric mean is used, the new coefficient is $\sqrt[3]{d \cdot d^2 \cdot d^3} = d^2$.

Any aggregation scheme introduces a bias. But a scheme that more accurately approximates the "true" coefficient is desirable. Therefore we will use an aggregation scheme that computes the geometric mean of original coefficients for those columns in the CESM and IESM portion that display the exponentially declining coefficients.
B. Geometric aggregation scheme

An outline of the geometric scheme follows.

1. Take the geometric mean of the column's coefficients across all rows of the periods to be aggregated.

2. Add coefficients across columns of the periods to be aggregated.

Columns in the CESM and IESM other than these capacity columns are aggregated in the standard aggregation scheme.

The following is an example of aggregation of 3 periods with a representative survival rate of 0.6 and an increasing service need.

Without aggregation, the solutions are

\[ x_1 = y, \quad x_2 = 1.4y, \quad x_3 = 1.8y \]

A representative value, the mean of \( x_1, x_2, x_3 \) is \( \bar{x} = 1.4y \) for the new period 1.

Using the geometric aggregation scheme, the single resulting equation and its solution are:
Using the arithmetic aggregation scheme, the single resulting equation and its solution are

\[ 4.35x = 6y, \quad \bar{g} = 1.38 \]

\[ 4.56x = 6y, \quad \bar{a} = 1.32 \]

This result illustrates that geometric aggregation gives results closer to the original 5 year period model.

C. Modification of variable time model program

The arithmetic aggregation scheme is implemented in a FORTRAN program that processes the MPS - format LP matrix listing. As a programming convenience in the first implementation, only additions are made and the coefficients in arithmetic aggregation are not divided by the number of periods in the aggregation. Thus two identical rows would appear aggregated as one row, but with all coefficients multiplied by two. The geometric aggregation scheme must therefore multiply the geometric mean by the number of periods to maintain correct linkage and consistency with the rest of the model.

The aggregation takes place in two stages, first across rows then across columns. Modification to the existing program is done in two parts. The first is shown in Figure 1 where row aggregation in the main program is done. The second is shown in Figure 2 where column aggregation is done in the subroutine UPDATE. To distinguish the two aggregation modes, marker cards reading "*ARITH" and "*GEOM" are needed in the input deck. If no marker appears, the program defaults to arithmetic aggregation for the entire matrix.
Figure 1

Row aggregation in the MAIN program

ARITHMETIC AGGREGATION
Give $\sum$ coefficients across rows

To UPDATE

* GEOM ?

YES

Calculate $\Pi$ coefficients across rows

To UPDATE

NO

* ARITH ?

YES
Figure 2

Column aggregation in the subroutine UPDATE

ARITH? YES -> add across columns

NO ->

take

$(\# \text{ periods}) \times \left\{ \prod \text{ coeffs} \right\}^{1/\#\text{ periods}}$

add across columns

RETURN To MAIN
III. Test Run of Modified Variable Time Model

A. Test run

Test run of the geometric aggregation scheme was made and compared with arithmetic aggregation results. To minimize computational costs the development and testing was done on an eight period 40 year model which contained the CESM but not the IESM. The eight periods were aggregated to five planning periods covering the same horizon. The qualitative results presented here will generalize both to a longer time horizons and to a model containing the IESM.

The test runs were made with an unaggregated 8 period "Longdeck" and two aggregated 5 period models: "Short G", derived using the modified geometric aggregation scheme and "Short A", derived using only the arithmetic aggregation. A single aggregation mapping of periods from Longdeck to either aggregated matrix was tested. The mapping 1-2-3-1-1 yields 5 periods in the short deck with lengths 5, 10, 15, 5 and 5. For example periods 2 and 3 from longdeck become period 2 in either short A or short G. No aggregation is done for either the first or last periods in order that the aggregated matrices have several periods with identical coefficients as the original.

B. Comparison of results

The observed objective function values were

<table>
<thead>
<tr>
<th></th>
<th>Longdeck</th>
<th>Short G</th>
<th>Short A</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5745.16663</td>
<td>5701.71558</td>
<td>5654.43644</td>
</tr>
</tbody>
</table>

8
Note that the objective value of Short G is closer than that of Short A to the value of the unaggregated Longdeck.

The comparison of objective values alone is not sufficient to indicate that Short G yields better results. We will also present comparison of more detailed model activities. Gross National Product and Total Primary Energy Consumption are activities from the main portion of the model. Their aggregation is done using an arithmetic scheme in both the original and modified variable time programs. The results given in Table 1 show that differences are small between aggregation schemes for these two variables. However, we note that the numerical values from the modified program are larger than those from the original program.

The modified aggregation scheme focused on the CESM portion of the integrated PILOT model. The CESM models use four energy services; space heat, other thermal residential, air conditioning, and automobile drive. A total of 55 energy service technologies provide these four services. Due to the structure of the CESM we cannot expect the LP solution of an aggregated model to agree with the unaggregated solution in all 55 technologies for every period. However, the totals across energy service types demonstrate that a geometric aggregation scheme for CESM capital stock variables yields solution closer to the unaggregated values. Tables 2 and 3 present the solution values for all technologies in two of the four CESM energy services. The survival rates are 0.918 for Space Heat and 0.59 for Other Thermal.
### Table 1

GNP and Total Primary Energy Consumption Comparison

<table>
<thead>
<tr>
<th>activity</th>
<th>GNP</th>
<th>Total Primary Energy Consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>old period scheme</td>
<td>1906.110</td>
<td>2346.653</td>
</tr>
<tr>
<td>new period scheme</td>
<td>2545.416</td>
<td>3646.218</td>
</tr>
<tr>
<td>Short C</td>
<td>1906.109</td>
<td>2561.071</td>
</tr>
<tr>
<td>Short A</td>
<td>1906.109</td>
<td>2513.325</td>
</tr>
<tr>
<td>Longdeck</td>
<td>75.201</td>
<td>82.395</td>
</tr>
<tr>
<td>Short C</td>
<td>75.504</td>
<td>90.279</td>
</tr>
<tr>
<td>Short A</td>
<td>75.504</td>
<td>87.974</td>
</tr>
</tbody>
</table>
### Table 2
Space Heat Comparison

<table>
<thead>
<tr>
<th>Energy Service</th>
<th>old period used scheme</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERO</td>
<td>Longdeck</td>
<td>.302</td>
<td>.286</td>
<td>.268</td>
<td>.249</td>
<td>.228</td>
<td>.205</td>
<td>.179</td>
<td>.151</td>
</tr>
<tr>
<td></td>
<td>Short G</td>
<td>.302</td>
<td>.277</td>
<td></td>
<td>.227</td>
<td>.179</td>
<td>.151</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Short A</td>
<td>.302</td>
<td>.277</td>
<td></td>
<td>.227</td>
<td>.179</td>
<td>.151</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ER2</td>
<td>L</td>
<td>.019</td>
<td>.119</td>
<td>.362</td>
<td>.738</td>
<td>1.327</td>
<td>1.851</td>
<td>2.659</td>
<td>3.441</td>
</tr>
<tr>
<td></td>
<td>S G</td>
<td>.019</td>
<td>.194</td>
<td></td>
<td>.784</td>
<td></td>
<td>1.621</td>
<td>2.488</td>
<td></td>
</tr>
<tr>
<td></td>
<td>S A</td>
<td></td>
<td>.081</td>
<td>.065</td>
<td>.055</td>
<td>.051</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HP2</td>
<td>L</td>
<td>.012</td>
<td>.011</td>
<td>.010</td>
<td>.009</td>
<td>.008</td>
<td>.007</td>
<td>.006</td>
<td></td>
</tr>
<tr>
<td></td>
<td>S G</td>
<td>.012</td>
<td>.120</td>
<td></td>
<td>.096</td>
<td></td>
<td>.081</td>
<td>.075</td>
<td></td>
</tr>
<tr>
<td></td>
<td>S A</td>
<td>.012</td>
<td>.011</td>
<td></td>
<td>.009</td>
<td></td>
<td>.007</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SE1</td>
<td>L</td>
<td>.019</td>
<td>.046</td>
<td>.295</td>
<td>.676</td>
<td>.666</td>
<td>.612</td>
<td>.755</td>
<td>.693</td>
</tr>
<tr>
<td></td>
<td>S G</td>
<td>.019</td>
<td>.194</td>
<td></td>
<td>.310</td>
<td></td>
<td>.504</td>
<td>.553</td>
<td></td>
</tr>
<tr>
<td></td>
<td>S A</td>
<td></td>
<td>.019</td>
<td>.194</td>
<td></td>
<td>.156</td>
<td></td>
<td>.564</td>
<td>.830</td>
</tr>
<tr>
<td>SE2</td>
<td>L</td>
<td>1.493</td>
<td>1.412</td>
<td>1.325</td>
<td>1.229</td>
<td>1.124</td>
<td>1.010</td>
<td>.884</td>
<td>.748</td>
</tr>
<tr>
<td></td>
<td>S G</td>
<td>1.493</td>
<td>1.369</td>
<td></td>
<td>1.121</td>
<td></td>
<td>.884</td>
<td>.748</td>
<td></td>
</tr>
<tr>
<td></td>
<td>S A</td>
<td>1.493</td>
<td>1.369</td>
<td></td>
<td>1.121</td>
<td></td>
<td>.884</td>
<td>.748</td>
<td></td>
</tr>
<tr>
<td>FG0</td>
<td>L</td>
<td>.019</td>
<td>.017</td>
<td>.194</td>
<td>.319</td>
<td>.293</td>
<td>.269</td>
<td>.247</td>
<td>.227</td>
</tr>
<tr>
<td></td>
<td>S G</td>
<td>.019</td>
<td>.114</td>
<td></td>
<td>.092</td>
<td></td>
<td>.078</td>
<td>.071</td>
<td></td>
</tr>
<tr>
<td></td>
<td>S A</td>
<td>.019</td>
<td>.113</td>
<td></td>
<td>.091</td>
<td></td>
<td>.077</td>
<td>.070</td>
<td></td>
</tr>
<tr>
<td>SG2</td>
<td>L</td>
<td>1.171</td>
<td>1.108</td>
<td>1.039</td>
<td>.964</td>
<td>.882</td>
<td>.792</td>
<td>.694</td>
<td>.586</td>
</tr>
<tr>
<td></td>
<td>S G</td>
<td>1.171</td>
<td>1.074</td>
<td></td>
<td>.879</td>
<td></td>
<td>.694</td>
<td>.586</td>
<td></td>
</tr>
<tr>
<td></td>
<td>S A</td>
<td>1.171</td>
<td>1.074</td>
<td></td>
<td>.879</td>
<td></td>
<td>.694</td>
<td>.586</td>
<td></td>
</tr>
<tr>
<td>F00</td>
<td>L</td>
<td>.411</td>
<td>.377</td>
<td>.346</td>
<td>.318</td>
<td>.292</td>
<td>.268</td>
<td>.246</td>
<td>.226</td>
</tr>
<tr>
<td></td>
<td>S G</td>
<td>.411</td>
<td>.376</td>
<td></td>
<td>.304</td>
<td></td>
<td>.256</td>
<td>.235</td>
<td></td>
</tr>
<tr>
<td></td>
<td>S A</td>
<td>.411</td>
<td>.361</td>
<td></td>
<td>.292</td>
<td></td>
<td>.246</td>
<td>.226</td>
<td></td>
</tr>
<tr>
<td>F02</td>
<td>Longdeck</td>
<td>3.446</td>
<td>3.6235</td>
<td>4.834</td>
<td>5.717</td>
<td>6.379</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Short G</td>
<td>3.446</td>
<td>3.718</td>
<td>3.813</td>
<td>4.297</td>
<td>4.907</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Short A</td>
<td>3.446</td>
<td>3.674</td>
<td>3.624</td>
<td>4.327</td>
<td>5.157</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 3
Other Thermal Comparison

<table>
<thead>
<tr>
<th>Energy Service</th>
<th>Old period</th>
<th>New period</th>
<th>Energy used</th>
<th>scheme</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Longdeck</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Short G</td>
<td>.350</td>
<td>.132</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Short A</td>
<td>.350</td>
<td>.081</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EW0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EW1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S G</td>
<td>.609</td>
<td>.212</td>
<td>.125</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S A</td>
<td>.511</td>
<td>.178</td>
<td>.105</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EW2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S G</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EW3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S G</td>
<td>.460</td>
<td>.867</td>
<td>1.074</td>
<td>1.188</td>
<td>1.363</td>
<td>.804</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S A</td>
<td>.653</td>
<td>1.075</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SW0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S G</td>
<td>.741</td>
<td>.279</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S A</td>
<td>.741</td>
<td>.172</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SW1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>.168</td>
<td>.786</td>
<td>.625</td>
<td>.369</td>
<td>.218</td>
<td>.128</td>
<td>.076</td>
<td>.045</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S G</td>
<td>.168</td>
<td>.833</td>
<td>.222</td>
<td></td>
<td>.077</td>
<td>.046</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S A</td>
<td>.168</td>
<td>.802</td>
<td>.214</td>
<td></td>
<td>.075</td>
<td>.044</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SW2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S G</td>
<td>.013</td>
<td>.131</td>
<td>.035</td>
<td>.035</td>
<td>.292</td>
<td>.500</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S A</td>
<td>.013</td>
<td>.131</td>
<td>.035</td>
<td>.310</td>
<td>.523</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>1.271</td>
<td>1.342</td>
<td>1.736</td>
<td>2.624</td>
<td>1.528</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S G</td>
<td>1.271</td>
<td>1.252</td>
<td>1.137</td>
<td>1.328</td>
<td>1.800</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S A</td>
<td>1.271</td>
<td>1.216</td>
<td>0.995</td>
<td>1.360</td>
<td>1.862</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Note that solution values from the geometric aggregation are greater than the those from the arithmetic aggregation. This is a general result and can be stated as the following proposition.

Proposition

\[ \bar{x}_g \geq \bar{x}_a \]

where \( x_i \) has exponentially declining coefficients.

Proof: For \( 0 < d < 1 \), the geometric mean of the powers of \( d \leq \) arithmetic mean, i.e.

\[
\left( \prod_{i=1}^{n} d^{i-1} \right)^{1/n} \leq \frac{1}{n} \sum_{i=1}^{n} d^{i-1}
\]

Therefore, the new coefficients of geometrically aggregated periods are less than or equal to the corresponding coefficients in arithmetically aggregated periods.

The energy service demanded in PILOT is influenced indirectly by the total investments in energy facility capital stocks, but this influence is quite small and not large enough to overcome the difference in coefficient values between the aggregation schemes. Therefore, the integrated solutions of the aggregated models will exhibit similar values for GNP and other macroeconomic values and larger values for total CESM capital stocks in the modified aggregation.

C. Conclusion

Any aggregation scheme introduces some aggregation bias, which indicates information is lost. For the PILOT CESM and IESM, this aggregation
bias can be reduced by using a scheme based on a geometric mean of coefficients. The numerical results and the proposition above show that CESM and IESM values from a geometric aggregation are larger in absolute values than those from an arithmetic scheme.

The capital stock structure of the CESM embodies information of two types. Stocks are installed that provide energy service demand within a single time period and that replace earlier vintages of capital stock that have depreciated. This inter-temporal depreciation relation is destroyed by aggregation. By using a geometric aggregation scheme, solution values more closely approximate the representative values from the unaggregated periods, thus recapturing some lost information and reducing aggregation bias. Even though numerical results are presented for a short time horizon in a model containing only the CESM, the qualitative results are expected to hold for longer time horizons and for an integrated model containing the IESM as well.
REFERENCES


1. // JOB ,CLASS=E, REGION=256K, TIME=(10,00)
2. //
3. // VARIABLE TIME PERIOD PROGRAM
4. //
5. //MAIN HOLD=OUTPUT
6. //DELCONDS EXEC PGM=IEFBRI4
7. //DD DSN=WYL.WJ.***.SHORT, VOL=SER=WORK03, UNIT=DISK,
8. //DISP=(OLD,DELETE)
9. //
10. // THE PRECEDING STEP (DELCONDS) SHOULD DELETE THE OUTPUT FILE
11. // FROM ANY PREVIOUS RUN OF THIS PROGRAM
12. //
13. // VARIABLE TIME PERIOD PROGRAM
14. //
15. // INPUT - TWO CARDS LOCATED AT END OF THIS DECK
16. // (2ND CARD CONTAINS AGGREGATION SCHEME)
17. //
18. // OUTPUT - FTO6F001 LIST OF WARNING MESSAGES
19. // (SEE COMMENTS WITHIN PROGRAM FOR ASSUMPTIONS)
20. //
21. // DOCUMENTATION - COMMENTS WITHIN THIS PROGRAM
22. //
23. // MPS III MATH. PROG. SYSTEM USER MANUAL, SECTION 6,
24. //
25. // ENERGY PROJECT MEMO # 76-34, "THE PROCEDURE OF USING THE
27. //
28. // ENERGY PROJECT MEMO # 77-1, "AGGREGATION OF CONSTRAINTS AND
29. // VARIABLES IN LINEAR PROGRAMS", RICHARD WOLLMER, JAN. 1977.
30. //
31. // ENERGY PROJECT MEMO # 77-19, "ANALYSIS OVER LONGER PLANNING
32. // HORIZON IN THE PILOT ENERGY/ECONOMIC MODEL", NATHAN
34. //
35. // SOL WORKING PAPER #76-3, "VARIABLE-TIME PERIODS AND END-
36. // CONDITION EFFECTS OF THE PILOT ENERGY MODEL",
38. //
39. // EXEC WATFIV,FORVER=NEW
40. // FTO6F001 DD SYSOUT=A
41. // FTO5F001 DD UNIT=DISK, DSN=WYL.WJ.***.LONGDECK, VOL=SER=WORK03,
42. // DISP=SHR
43. // FTO9F001 DD UNIT=DISK, DSN=WYL.WJ.***.SHORT, VOL=SER=WORK03,
44. // SPACE=(TRK,(200,20), RLSE), DISP=(NEW,KEEP),
45. // DCB=(RECFM=FB, LRECL=80, BLKSIZE=3120)
46. // GO.SYSIN DD *
47. $WATFIV
49. CHARACTER*1 PRNAME(100,2),TYPIN(4),NAMEIN(8),NEWNAME(8), ROWNAME(8),COLNAME(8),CHNAME(8),RNAME(2,8),
50. 1 PRNAME(100,2),TYPIN(4),TYPE(4),NAIN(8),PNEWNAME(8),
51. 1 ROWNAME(8),COLNAME(8),CNAME(8),RNAME(2,8),
52. 2 COL(4),RHS(4),BOUNDS(4),ENDATA(4),GEM(4),ARI(4),
53. 2 FR(2),FX(2),UP(2),LO(2),MI(2),
54. CHARACTER*50 CARD
55. INTEGER PB,PLISTIN(20),INOUT(20)
56. LOGICAL PASSI,ARITH,GEOM,MINUSI,TRUE,FALSE
57. C---
58. C--- (THE DIMENSION OF NAMETAB AND VALUTAB SHOULD EXCEED THE MAXIMUM
59. C--- NUMBER OF MPS ROW ENTRIES ANTICIPATED IN ANY COLUMN, AFTER
60. C--- AGGREGATION. ANY REDIMENSIONING MUST BE CARRIED OUT IN SUBROUTINES
61. C--- UPDATE AND COLOUT AS WELL)
62. C---
63. CHARACTER*1 NAMETAB(100,8)
64. DIMENSION VALUTAB(100),VALUTAB(100)
65. COMMON/BLOCK1/NPRIN,NPROUT,INOUT,PRNAME,PRNAME
66. COMMON/BLOCK2/COLNAME,NAMETAB,VALUTAB,MAXENT
67. C---
68. C---
69. C---
70. C---
71. C INITIALIZE CONSTANTS AND READ AGGREGATION SCHEME
72. C---
73. C--- INITIALIZE ARRAY DIGIT TO CHARACTER EQUIVALENTS OF THE 10 DIGITS
74. C---
75. DATA DIGIT(1),DIGIT(2),DIGIT(3),DIGIT(4), DIGIT(5),DIGIT(6),DIGIT(7),DIGIT(8)/,
76. DATA DIGIT(9),DIGIT(10),DIGIT(11)/,
77. C---
78. C--- INITIALIZE COL,RHS,BOUNDS, AND ENDATA (MPS SEGMENT NAMES)
79. C---
80. DATA COL(1),COL(2),COL(3),COL(4)/,
81. DATA RHS(1),RHS(2),RHS(3),RHS(4)/,
82. DATA BOUNDS(1),BOUNDS(2),BOUNDS(3)/,
83. DATA ENDTA(1),ENDTA(2),ENDTA(3)/,
84.1 DATA GEM(1),GEM(2),GEM(3),GEM(4)/,
85.2 DATA ARI(1),ARI(2),ARI(3),ARI(4)/,
86. C---
87. C--- INITIALIZE FR,FX,UP,LO AND MI (BOUND TYPE NAMES)
88. C---
89. DATA FR(1),FR(2),FX(1),FX(2)/,
90. DATA UP(1),UP(2),LO(1),LO(2)/,
91. C---
92. C--- INITIALIZE BLANK AND ASTERSK (SINGLE CHARACTERS)
93. C---
94. DATA BLANK,ASTERSK/ ,
95. C---
96. C--- INITIALIZE TRUE AND FALSE (LOGICALS)
97. C---
98. DATA TRUE,FALSE,/
INITIALIZE 2-DIM. ARRAY, PRNAME, TO TWO CHARACTER EQUIVALENTS
OF EACH POSSIBLE PERIOD NUMBER (I.E. ('0','0') TO ('9','9'))
LOOP OVER I TO SELECT FIRST DIGIT
J TO SELECT SECOND DIGIT
DO 20 I=1,10
DO 20 J=1,10
(N-TH LEVEL OF ARRAY PRNAME CORRESPONDS TO N-TH POSITION
IN THE SEQUENCE 00,01,02,03,...,99)
N = (I-1)*10 + J
PRNAME(N,1) = DIGIT(I)
PRNAME(N,2) = DIGIT(J)
EXAMPLES -
09 IS IN 10-TH POSITION
N=10 RESULTS FROM I=1,J=0
10 IS IN 11-TH POSITION
N=11 RESULTS FROM I=2,J=1
I=1 SELECTS DIGIT '0'
J=1 SELECTS DIGIT '0'
I=2 SELECTS DIGIT '1'
J=10 SELECTS DIGIT '9'
CONTINUE
INITIALIZE MAXPR (MAXIMUM NUMBER OF PERIODS IN OUTPUT MODEL)
MAXPR = 20
INITIALIZE MAXENT (MAXIMUM NUMBER OF ENTRIES IN OUTPUT TABLES)
AND LASTIX (INDEX OF LAST ENTRY IN OUTPUT TABLES)
MAXENT = 100
LASTIX = 0
(TWO INPUT CARDS ARE LOCATED AT THE END OF THIS DECK)
READ AND IGNORE DUMMY CARD (USED ONLY FOR IDENTIFYING FIELDS
OF NEXT CARD WHEN KEYING IN DATA)
READ (5,900) CARD
900 FORMAT (A80)
READ AGGREGATION SCHEME CARD
READ (5,902) LISTIN
902 FORMAT(20I3)
(LISTIN - THE I-TH NUMBER IN LISTIN IS THE NUMBER OF PERIODS
FROM THE INPUT MODEL TO BE AGGREGATED WHEN FORMING
The I-TH PERIOD OF THE OUTPUT MODEL)
COMPUTE NPROUT (NUMBER OF PERIODS IN OUTPUT MODEL
= NUMBER OF NONZEROS IN LISTIN)
151. C---
152. DO 40 N=1,MAXPR
153. IF (LISTIN(N).EQ.0) GO TO 50
154. 40 CONTINUE
155. C--- NO ZEROS IN LISTIN.
156. C--- N EQUALS NUMBER OF NONZEROS IN LISTIN PLUS ONE.
157. C---
158. 50 NPROUT = N - 1
159. IF (NPROUT.EQ.0) STOP
161. C---
162. LASTPR = 0
163. DO 60 I=1,MAXPR
164. LASTPR = LASTPR + LISTIN(I)
165. INOUT(I) = LASTPR
166. 60 CONTINUE
167. C--- SAVE NPRIN (NUMBER OF PERIODS IN THE INPUT MODEL)
168. C--- COMMON BLOCK1 IS NOW WELL-DEFINED.
169. C--- PRINT OUT AGGREGATION SCHEME
170. WRITE (6,991)
171. 991 FORMAT (I1,'INPUT CARDS--')
172. WRITE (6,992) CARD
173. 992 FORMAT (I1,'A80')
174. WRITE (6,993) LISTIN
175. 993 FORMAT (I1,'2013')
176. WRITE (6,994)
177. 994 FORMAT (I1,'LIST OF LAST INPUT PERIOD NUMBER CORRESPONDING TO EACH OUTPUT PERIOD NUMBER--')
178. WRITE (6,995) CARD
179. 995 FORMAT (I1,'A80')
180. WRITE (6,996) (INOUT(I),I=1,NPROUT)
181. 996 FORMAT (I1,'2013')
182. WRITE (6,997)
183. 997 FORMAT (I1,'LIST OF WARNING MESSAGES - SEE COMMENTS WITHIN ',
184. 'PROGRAM FOR DEFAULT ACTION--')
203. C SWITCH TO INPUT MODEL FILE
204. C---
205. C--- COPY "NAME" CARD AND "ROWS" CARD TO OUTPUT FILE
206. C---
207. DO 3 I=1,2
208. READ (8,900) CARD
209. WRITE (9,900) CARD
210. 80 CONTINUE
211. C---
212. C---
213. C---
214. C---
215. C---
216. C ROWS SEGMENT BEGINS
217. C---
218. C--- BEGIN ROWS CARD CYCLE - ONE PASS FOR EACH CARD READ
219. C---
220. C--- SET PASSI (FIRST PASS INDICATOR) ON OR OFF
221. C---
222. PASSI = TRUE
223. GO TO 105
224. 100 PASSI = FALSE
225. C---
226. C--- READ A CARD USING ROW CAPD FORMAT
227. C---
228. 105 READ (6,910) TYPIN,NAMEN
229. 910 FORMAT (4AI,0A1)
230. C---
231. C--- (TYPIN - ROW TYPE)
232. C--- (NAMEN - ROW NAME)
233. C---
234. C--- SKIP COMMENT CARDS
235. C---
236. IF (TYPIN(1).EQ.ASTERSK) GO TO 105
237. C---
238. C--- REFORM ROW NAME BY CALLING SUBROUTINE RENAME
239. C---
240. C--- CALL RENAME(NAMEN,NENNAME,INEWR,NBLANK)
241. C---
242. C--- (NENNAME - NAME OF INPUT ROW AS IT IS TO APPEAR ON OUTPUT)
243. C--- (NBLANK - NUMBER OF BLANKS AT END OF NENNAME)
244. C---
245. C--- ON FIRST PASS, BRANCH TO "NEW ROW NAME"
246. C---
247. IF (PASSI) GO TO 170
248. C---
249. C--- (ASSUME MPS INPUT FILE SORTED SO THAT ROW NAMES WITH THE
250. C--- SAME ROOT ARE GROUPED TOGETHER IN ASCENDING ORDER OF PERIOD
251. C--- NUMBER, CONSEQUENTLY, AFTER NAMES ARE REFORMED THE ROWS
252. C--- WHICH ARE TO BE AGGREGATED WILL BE GROUPED TOGETHER UNDER
253. C--- A COMMON (OUTPUT) ROW NAME)
254. C---
255. C--- COMPARE NEWNAME WITH ROWNAME (THE NEWNAME OF THE PREVIOUS CARD)
256. C--- (TO SAVE TIME, CONSIDER ONLY NONBLANK CHARACTERS OF NEWNAME)
257. C---
258. C--- NONBLK = 8 - NBILANK
259. DO 110 I=1,NONBLK
260. C---
261. C--- IF NO MATCH, BRANCH TO "OUTPUT PREVIOUS ROW"
262. C---
263. C--- IF (NEWNAME(I).NE.ROWNAME(I)) GO TO 150
264. 110 CONTINUE
265. C---
266. C--- NAMES MATCH. (ROW ID FOR NEWNAME HAS ALREADY BEEN SET UP)
267. C--- PRINT WARNING IF TYPE DIFFERS FROM TYPE OF PREVIOUS CARD, THEN
268. C--- GO READ A NEW ROWS CARD
269. C---
270. DO 120 I=1,2
271. C---
272. 120 CONTINUE
273. GO TO 100
274. 130 WRITE (6,950) TYPIN,NAMEIN,TYPE,ROWNAME
275. 950 FORMAT (IH,'ROW ID INPUT AS ',4A1,8A1,' WILL BE OUTPUT AS ',
276. 1 4A1,8A1,' ** TYPE CHANGE')
277. GO TO 100
278. C---
279. C--- OUTPUT PREVIOUS ROW ID USING ROW CARD FORMAT
280. C---
281. 150 WRITE (9,910) TYPE,ROWNAME
282. C---
283. C--- IF NAMES DID NOT MATCH BECAUSE NEW CARD WAS "COLUMNS" CARD
284. C--- BRANCH TO "COLUMN SEGMENT BEGINS"
285. C---
286. DO 160 I=1,4
287. C---
288. 160 CONTINUE
289. DO 190 I=1,4
290. TYPE(I) = TYPIN(I)
291. 190 CONTINUE
292. C---
293. C--- GO READ A NEW ROWS CARD
C--- COLUMN SEGMENT BEGINS
C--- OUTPUT "COLUMNS" CARD
C---
200 WRITE (9,912)
912 FORMAT ('COLUMNS')
COUNT=1
ARITH=TRUE
GEOM=FALSE
MINUS=FALSE
C--- BEGIN COLUMN CARD CYCLE - ONE PASS FOR EACH CARD READ
C--- SET PASSI (FIRST PASS INDICATOR) ON OR OFF
C---
PASSI = TRUE
GO TO 210
PASSI = FALSE
GO TO 210
C---
C--- READ A CARD USING COLUMN CARD FORMAT
C---
207 DO 209 J=2,4
IF (TYPIN(J).N.E.ARI(J)) GO TO 210
CONTINUE
GEOM=FALSE
READ (8,914) TYPIN, CHNAME, (RNAME(I, J), I=1,8), RVALU(1),
(2) (RNAME(2, J), J=1,6), RVALU(2)
914 FORMAT (4A1, 8A1, 2X, 2(8A, 2X, F12.6, 3X))
C---
C--- (TYPIN = BLANK)
C--- (CHNAME = COLUMN NAME)
C--- (RNAME(1) AND (2) = ROW NAMES OF MATRIX ENTRIES)
C--- (RVALU(1) AND (2) = MATRIX ENTRIES)
C---
C--- SKIP COMMENT CARDS
C---
IF (TYPIN(1).NE.ASTERSK) GO TO 215
DO 229 J=2,4
IF (TYPIN(J).NE.GEM(J)) GO TO 207
CONTINUE
GEOM=TRUE
GO TO 210
C---
C--- REFORM COLUMN NAME BY CALLING SUBROUTINE RENAME
345. C---
346. 215 CALL RENAME(CNAME,NEWNAME,NEWPR,NBLANK)
347. C---
348. C--- (NEWNAME - NAME OF INPUT COLUMN AS IT IS TO APPEAR ON OUTPUT)
349. C--- (NEWNAME - NUMBER OF BLANKS ON END OF NEWNAME)
350. C---
351. C--- ON FIRST PASS, BRANCH TO "NEW COLUMN"
352. C---
353. C---
354. C--- IF (PASS1) GO TO 250
355. C---
356. C--- (ASSUME MPS INPUT FILE IS SORTED SO THAT COLUMN NAMING WITH
357. C--- THE SAME ROOT ARE GROUPED TOGETHER IN ASCENDING ORDER OF
358. C--- PERIOD NUMBER. CONSEQUENTLY, AFTER NAMES HAVE BEEN REFORMED
359. C--- THE COLUMNS WHICH ARE TO BE AGGREGATED WILL BE GROUPED
360. C--- TOGETHER UNDER A COMMON (OUTPUT) COLUMN NAME)
361. C---
362. C--- IF NEWNAME IS ALL BLANKS BRANCH TO "UPDATE OUTPUT TABLES"
363. C--- (ASSUME NEW CARD WAS "RHS" CARD)
364. C---
365. C---
366. C--- COMPARE NEWNAME WITH COLNAME (THE NEWNAME OF THE PREVIOUS CARD)
367. C--- (CONSIDER ONLY NONSLANK CHARACTERS OF NEWNAME)
368. C---
369. NONBLK = 8 - NBLANK
370. DO 220 I=1,NONBLK
371. C---
372. C--- IF NO MATCH, BRANCH TO "UPDATE OUTPUT TABLES"
373. C---
374. C--- IF (NEWNAME(I).NE.COLNAME(I)) GO TO 230
375. 220 CONTINUE
376. C---
377. C--- NAMES MATCH. OUTPUT TABLES FOR THIS COLUMN HAVE ALREADY BEEN
378. C--- SET UP.
379. C--- BRANCH TO "PROCESS ROW NAMES AND VALUES FROM CURRENT CARD"
380. C---
381. GO TO 275
382. C---
383. C--- UPDATE OUTPUT TABLES FOR PREVIOUS COLUMN WITH
384. C--- (LASTIX - INDEX OF LAST ENTRY IN OUTPUT TABLES)
385. C--- ROHALME (LAST ROWNAME ENCONTERED) AND ROWVALU (ASSOCIATED
386. C--- MPS MATRIX ENTRY) BY CALLING SUBROUTINE UPDATE
387. C---
388. 230 CALL UPDATE(LASTIX,ROWNAME,ROWVALU,PBLANK,ARITH,GEOM,COUNT,
389. 1,MINUSI)
390. C---
391. C---
392. C--- (LASTIX - INDEX OF LAST ENTRY IN OUTPUT TABLES)
393. C---
394. C--- (PBLANK - NUMBER OF BLANKS ON END OF ROWNAME)
395. C--- (COMMON BLOCK2 SHOULD BE WELL-DEFINED AT THIS POINT)
396. C--- (COLNAME - OUTPUT NAME OF AGGREGATED COLUMN)
396. C--- (NAMETAB - LIST OF AGGREGATED ROW NAMES FOR THIS COLUMN)
397. C--- (VALUTAB - CORRESPONDING LIST OF AGGREGATED MPS MATRIX ENTRIES)
398. C--- (LASTIX - INDEX OF LAST ENTRY IN NAMETAB AND VALUTAB)
399. C---
400. CALL COLOUT(LASTIX)
401. C---
402. C--- IF NAMES DID NOT MATCH BECAUSE NEW CARD WAS "RHS" CARD,
403. C--- BRANCH TO "RHS SEGMENT BEGINS"
404. C---
405. DO 240 I=1,4
406. C--- IF TYPE NOT EQUAL TO 'R', 'H', 'S', ' ', BRANCH TO "NEW COLUMN"
407. C---
408. IF (TYPIN(I).NE.RHS(II)) GO TO 250
409. 240 CONTINUE
410. GO TO 400
411. C---
412. C--- NEW COLUMN ENCOUNTERED.
413. C---
414. C--- ERASE OUTPUT TABLES, NAMETAB AND VALUTAB (A SAFETY MEASURE)
415. C---
416. C---
417. 250 NERASE = LASTIX + 1
418. NERASE = MIN(MAXENT,NERASE)
419. DO 260 I=1,NERASE
420. C--- VALUTAB(I) = 0.
421. C--- DO 260 J=1,8
422. C--- NAMETAB(I,J) = BLANK
423. 260 CONTINUE
424. C---
425. C--- RESET LASTIX (INDEX OF LAST ENTRY IN OUTPUT TABLES) TO ZERO
426. C---
427. C--- LASTIX = 0
428. C---
429. C--- SAVE NEW COLUMN NAME AS COLNAME AND ERASE ROWNAME (PREVIOUS
430. C--- ROWNAME PROCESSED)
431. C---
432. DO 270 I=1,8
433. C--- COLNAME(I) = NEWNAME(I)
434. C--- ROWNAME(I) = BLANK
435. 270 CONTINUE
436. C---
437. C--- RESET PBLANK (NUMBER OF BLANKS AT END OF PREVIOUS ROWNAME)
438. C--- TO EIGHT AND
439. C--- ROWVALU (AGGREGATE MPS ENTRY FOR PREVIOUS ROW NAME) TO ZERO
440. C---
441. PBLANK = 8
442. ROWVALU = 0.
443. C---
444. C--- PROCESS ROW NAMES AND VALUES FROM CURRENT CARD
445. C---
446. C--- LOOP ONCE FOR EACH (OF TWO) ROW NAMES (LOOP OVER I)
ASSUME MPS INPUT FILE IS SORTED SO THAT, FOR EACH INPUT COLUMN, ROW NAMES WITH THE SAME ROOT ARE GROUPED TOGETHER IN ASCENDING ORDER OF PERIOD NUMBER. CONSEQUENTLY, AFTER ROW NAMES ARE REFORMED, THE ROWS WHICH ARE TO BE AGGREGATED FOR THAT INPUT COLUMN WILL BE GROUPED TOGETHER UNDER A COMMON (OUTPUT) ROW NAME. HOWEVER, SINCE SEVERAL INPUT COLUMNS MAY NEED TO BE AGGREGATED UNDER ONE COLUMN NAME IT IS NECESSARY TO MAINTAIN NAMETAB (TABLE OF (OUTPUT) ROW NAMES ENCOUNTERED FOR THAT (OUTPUT) COLUMN), AND VALUTAB (TABLE OF CORRESPONDING (AGGREGATE) MPS MATRIX ENTRIES)

MOVE NAME(I) (INPUT ROW NAME BEING PROCESSED) INTO NAMEIN CALL STATEMENT WILL NOT ACCEPT AN IMPLIED DO LOOP

DO 280 J=1,8 NAMEIN(J) = RNAME(I,J) CONTINUE

REFORM INPUT ROW NAME BY CALLING SUBROUTINE RENAME CALL RENAME(NAMEIN,NEWNAME,INEWPR,NBLANK)

IF INPUT ROW NAME WAS ALL BLANKS, SKIP TO "END OF LOOP"

IF (NBLANK.EQ.8) GO TO 370

IF PREVIOUS ROW NAME ALL BLANKS, BRANCH TO "NEW ROW NAME"

IF (PBLANK.EQ.B) GO TO 350

COMPARE NEWNAME WITH ROWNAME (PREVIOUS ROWNAME PROCESSED) (CONSIDER ONLY NONBLANK CHARACTERS)

NONBLK = 8 - NBLANK

DO 285 J=1,NONBLK

IF NO MATCH, BRANCH TO "UPDATE OUTPUT TABLES"

IF (NEWNAME(J).NE.ROWNAME(J)) GO TO 290

CONTINUE

IF (ARITH) GO TO 288

IF (RVALU(I).LT.0.0) GO TO 287

ROWVALU=ROWVALU+RVALU(I)

COUNT=COUNT+1

GO TO 370

ROWVALU=ROWVALU+ABS(RVALU(I))

GO TO 286
493. C--- NAMES MATCH.
494. C--- ADD RVALU (CURRENT MATRIX ENTRY) TO ROWVALU (PREVIOUS TOTAL)
495. C--- AND BRANCH TO "END OF LOOP"
496. C---
497. 288 RONVALU = RONVALU + RVALU(I)
498. GO TO 370
499. C---
500. C--- UPDATE OUTPUT TABLES WITH ROWNAME (PREVIOUS ROW NAME)
501. C--- AND ROWVALU (CORRESPONDING MATRIX ENTRY) BY CALLING
502. C--- SUBROUTINE UPDATE
503. C---
504. 290 CALL UPDATE(LASTIX, ROWNAME, ROWVALU, PBLANK, ARITH, GEOM, COUNT, MINUS1)
505. C---
506. C--- NEW ROWNAME ENCOUNTERED.
507. C--- SAVE NEWNAME (CURRENT ROW NAME) AS ROWNAME (PREVIOUS ROW NAME)
508. C--- SAVE NBLANK (NUMBER OF BLANKS IN NEWNAME) AS PBLANK
509. C--- SAVE RVALU (CORRESPONDING MATRIX ENTRY) AS ROWVALU
510. C---
511. 350 DO 360 L=1,8
512. ROWNAME(L) = NEWNAME(L)
513. CONTINUE
514. PBLANK = NBLANK
515. IF (ARITH) GO TO 364
516. IF (RVALU(I).LT.0.0) GO TO 365
517. 364 ROWVALU=RVALU(I)
518.31 MINUS1=FALSE
519.4 GO TO 370
520.6 365 ROWVALU=ABS(RVALU(I))
521.6 MINUS1=TRUE
522. C---
523. C--- END OF LOOP
524. C---
525. 370 CONTINUE
526. C---
527. C--- BOTH ROW NAMES FROM INPUT CARD HAVE NOW BEEN PROCESSED.
528. C--- GO READ ANOTHER COLUMN CARD
529. C---
530. C---
531. C---
532. C---
533. C---
534. C RHS SEGMENT BEGINS
535. C---
536. C--- OUTPUT "RHS" CARD
537. C---
538. 400 WRITE (9,916)
539. 916 FORMAT ('RHS')
540. C----
541. C---- RESET ROWNAME (PREVIOUS ROW NAME) TO BLANKS AND
542. C---- ROWVALU (ASSOCIATED MPS MATRIX ENTRY) TO ZERO AND
543. C---- PBLANK (NUMBER OF BLANKS ON END OF ROWNAME) TO EIGHT
544. C----
545. DO 405 I=1,8
546. ROWNAME(I) = BLANK
547. 405 CONTINUE
548. ROWVALU = 0.
549. PBLANK = 8
550. C----
551. C---- BEGIN RHS CARD CYCLE - ONE PASS FOR EACH CARD READ
552. C----
553. C---- SET PASSI (FIRST PASS INDICATOR) ON OR OFF
554. C----
555. PASSI = TRUE
556. GO TO 420
557. 410 PASSI = FALSE
558. C----
559. C---- READ A CARD USING COLUMN CARD FORMAT
560. C----
561. 420 READ (8,914) TYPIN,CNAME,(RNAME(1:1),I=1,8),RVALU(1),
562. 1 (RNAME(2:J),J=I,8),RVALU(2)
563. C----
564. C---- (TYPIN = BLANK)
565. C---- (CNAME = RHS NAME)
566. C---- (RNAME(1) AND (2) = ROW NAMES OF RHS ENTRIES)
567. C---- (RVALU(1) AND (2) = RHS VALUES)
568. C----
569. C---- ON FIRST PASS ONLY, SAVE CNAME AS COLNAME (NAME OF RHS)
570. C----
570.1 C----
571. IF (.NOT.PASSI) GO TO 440
572. DO 430 I=1,8
573. COLNAME(I) = CNAME(I)
574. 430 CONTINUE
574.1 DO 435 I=1,4
574.2 TYPE(I) = TYPIN(I)
574.3 435 CONTINUE
575. C----
576. C---- PROCESS ROW NAME AND VALUES FROM CURRENT CARD
577. C----
578. C---- LOOP ONCE FOR EACH (OF TWO) ROW NAMES (LOOP OVER I)
579. C----
580. 440 DO 530 I=1,2
581. C----
582. C---- (ASSUME MPS INPUT FILE SORTED SO THAT ROW NAMES WITH SAME
583. C---- ROOT ARE GROUPED TOGETHER IN ASCENDING ORDER OF PERIOD
584. C---- NUMBER. CONSEQUENTLY, AFTER ROW NAMES ARE REFORMED, THE
585. C---- ROWS WHICH ARE TO BE AGGREGATED ON THE RHS WILL BE
586. C --- GROUPED TOGETHER UNDER A COMMON (OUTPUT) ROW NAME
587. C ---
588. C --- MOVE RNAME(I) (INPUT ROW NAME BEING PROCESSED) INTO NAMEIN
589. C --- (CALL STATEMENT WILL NOT ACCEPT AN IMPLIED DO LOOP)
590. C ---
591. DO 445 J=1,8
592. NAMEIN(J) = RNAME(I,J)
593. 445 CONTINUE
594. C ---
595. C --- MOVE PNAME(I) (INPUT ROW NAME BEING PROCESSED) INTO
596. C --- (CALL STATEMENT WILL NOT ACCEPT AN IMPLIED DO LOOP)
597. C ---
598. CALL RENAME(NAMEIN,NEWNAME,INEWPR,NBLANK)
599. C --- (NEWNAME - NAME OF INPUT ROW AS IT IS TO APPEAR ON OUTPUT)
600. C --- (NBLANK - NUMBER OF BLANKS AT END OF NEWNAME)
601. C ---
602. C --- IF PREVIOUS ROW NAME ALL BLANKS (I.E. FIRST NAME ON FIRST CARD)
603. C --- BRANCH TO "NEW ROW NAME"
604. C ---
605. C --- IF (PBLANK.EQ.8) GO TO 475
606. C ---
607. C --- IF INPUT ROW NAME NOT ALL BLANKS
608. C --- BRANCH TO "COMPARE NEWNAME WITH ROWNAME"
609. C ---
610. C --- IF (NBLANK.LT.8) GO TO 450
611. C ---
612. C --- IF INPUT ROW NAME NOT ALL BLANKS
613. C --- IF THIS IS FIRST NAME ON CARD ASSUME IT IS "BOUNDS" CARD AND
614. C --- BRANCH TO "NEW ROW NAME"
615. C --- OTHERWISE BRANCH TO "END OF LOOP"
616. C ---
617. C --- IF (I.EQ.1) GO TO 470
618. C --- GO TO 530
619. C ---
620. C --- COMPARE NEWNAME WITH ROWNAME (PREVIOUS ROW NAME)
621. C --- (CONSIDER ONLY NONBLANK CHARACTERS OF NEWNAME)
622. C ---
623. 450 NONBLK = 8 - NBLANK
624. DO 460 L=1,NONBLK
625. C ---
626. C --- IF NO MATCH, BRANCH TO "OUTPUT PREVIOUS ROW ENTRY"
627. C ---
628. C --- IF (NEWNAME(L).NE.ROWNAME(L)) GO TO 470
629. 460 CONTINUE
630. C ---
631. C --- NAMES MATCH.
632. C --- ADD CURRENT RHS ENTRY TO ROWVALU (PREVIOUS TOTAL) AND
633. C --- BRANCH TO "END OF LOOP"
634. C ---
635. ROWVALU = ROWVALU + RVALU(I)
636. C --- GO TO 530
637. C ---
OUTPUT PREVIOUS ROW ENTRY BY CALLING SUBROUTINE CARDOUT

CALL CARDOUT(TYPE, COLNAME, I, ROWNAME, ROWVALU, ROWNAME, ROWVALU)

(3-RO ARGUMENT IN CALL IS NUMBER OF ROW ENTRIES SUBMITTED
FOR OUTPUT - IN THIS CASE ONLY ONE SO THE 6-TH AND 7-TH
ARGUMENTS WILL BE IGNORED)

NEW ROW NAME ENCOUNTERED.

IF THIS IS FIRST NAME ON CARD CHECK IF "BOUNDS" CARD
OTHERWISE BRANCH TO "RESET OUTPUT BUFFERS"

IF (I.NE.1) GO TO 500

DO 480 L=1,4

IF TYPE NOT EQUAL TO 'B', 'O', 'U', 'H',
BRANCH TO "RESET OUTPUT BUFFERS"

IF (TYPIN(L).NE.BOUNDS(L)) GO TO 500

CONTINUE

GO TO 600

RESET OUTPUT BUFFERS - SAVE NEWNAME AS ROWNAME
- SAVE NBLANK AS P8LANK
- SAVE RVALU AS ROWVALU

DO 510 L=1,8
ROWVALU(L) = RVALU(I)
PBLANK = NBLANK
CONTINUE

END OF LOOP

CONTINUE

BOTH ROW NAMES FROM INPUT CARD HAVE NOW BEEN PROCESSED.
GO READ ANOTHER RHS CARD

GO TO 410

CONTINUE

BOUNDS SEGMENT BEGINS

OUTPUT "BOUNDS" CARD

WRITE (9,918)
FORMAT ("BOUNDS")
BEGIN BOUNDS CARD CYCLE - ONE PASS FOR EACH CARD READ

SET PASS1 (FIRST PASS INDICATOR) ON OR OFF

PASS1 = TRUE
GOTO 625
PASS1 = FALSE

READ A CARD USING BOUND CARD FORMAT

625 READ (8,920) TYPIN,(RNAME(I),I=1,8),CNAME,VALUE
FORMAT (4A1,8A1,2X,8A1,2X,F12.6)

(TYPIN - BOUND TYPE)
(RNAME(I) - BOUND NAME)
(CNAME - COLUMN NAME)
(VALUE - BOUND VALUE) (ASSUME ONLY ONE VALUE INPUT PER CARD)

SKIP COMMENT CARDS

IF (TYPIN(I).EQ.ASTERSK) GO TO 625

FIRST PASS ONLY, SAVE RNAME(I) AS ROWNAME (NAME OF BOUNDS "ROW")

IF (.NOT.PASS1) GO TO 640
DO 635 I=1,8
ROWNAME(I) = RNAME(I)
CONTINUE

CALL RENAME(CNAME,NEWNAME,INEWPR,NBLANK)

(NEWNAME - NAME OF INPUT COLUMN NAME AS IT IS TO APPEAR ON OUTPUT)
(INEWPR - INDEX OF OUTPUT PERIOD NUMBER)
(EQUALS ZERO IF CNAME DID NOT END WITH VALID INPUT PERIOD NUMBER)

IF FIRST PASS, BRANCH TO "NEW NAME/TYP"E

IF (PASS1) GO TO 700

ASSUME MPS INPUT FILE SORTED SO THAT FOR EACH BOUND TYPE ENCOUNTERED, COLUMN NAMES WITH THE SAME ROOT ARE GROUPED TOGETHER IN ASCENDING ORDER OF PERIOD NUMBER. CONSEQUENTLY, AFTER COLUMN NAMES ARE REFORMED, BOUNDS OF THE SAME TYPE WHICH ARE TO BE AGGREGATED WILL BE GROUPED TOGETHER UNDER A COMMON (OUTPUT) COLUMN NAME)

IF NEWNAME ALL BLANKS BRANCH TO "CHANGE IN BOUND NAME/TYP"
742. C--- (ASSUME "ENDATA" CARD ENCOUNTERED)
743. C--- IF (NBLANK.EQ.0) GO TO 670
744. C---
745. C--- COMPARE NEWNAME WITH COLNAME (THE NEWNAME OF THE PREVIOUS CARD)
746. C--- (CONSIDER ONLY THE NON-BLANK CHARACTERS OF NEWNAME)
747. C---
748. C--- NONBLK = 8 - NBLANK
749. DO 650 I=1,NONBLK
750. C---
751. C--- IF NO MATCH, BRANCH TO "CHANGE IN BOUND NAME/TYPE"
752. C---
753. C--- IF (NEWNAME(I).NE.COLNAME(I)) GO TO 670
754. C--- CONTINUE
755. C--- NAMES MATCH. NOW COMPARE BOUND TYPES
756. C---
757. DO 660 I=1,4
758. C--- IF (NEWNAME(I).NE.COLNAME(I)) GO TO 670
759. C--- CONTINUE
760. C--- NAMES AND BOUND TYPE MATCH.
761. C--- INCREMENT NBOUNDS (NUMBER OF BOUNDS ENCOUNTERED)
762. C--- ADD VALUE TO BDVALU (AGGREGATE BOUND VALUE)
763. C--- GO READ A NEW CARD
764. C---
765. C--- NBOUNDS = NBOUNDS + 1
766. BDVALU = BDVALU + VALUE
767. GO TO 620
768. C---
769. C--- CHANGE IN BOUND NAME/TYPE.
770. C--- IDENTIFY BOUND TYPE OF PREVIOUS NAME/TYPE
771. C---
772. 670 IF (TYPE(3).EQ.FR(2).OR.TYPE(3).EQ.MI(2)) GO TO 680
773. IF (TYPE(3).EQ.LO(2)) GO TO 685
774. IF (TYPE(3).EQ.FX(2).OR.TYPE(3).EQ.UP(2)) GO TO 690
775. C---
776. C--- TYPE HAS NOT ONE OF (MI,FR,FX,UP, OR LO).
777. C--- PRINT WARNING AND TREAT SAME AS "FREE" OR "MINUS INFINITY"
778. C---
779. WRITE 16,956) TYPE,RONAME,COLNAME,BOVALU
780. 956 FORMAT (1H ,4AA8A),2X,8A1,2X,F12.6,'** UNRECOGNIZED BOUND TYPE')
781. C---
782. C--- BOUND TYPE WAS EITHER "FREE" OR "MINUS INFINITY".
783. C--- OUTPUT BOUND CARD BY CALLING SUBROUTINE CARDOUT A
784. C---
785. C--- BRANCH TO "NEW NAME/TYPE"
786. C---
787. 680 CALL CARDOUT(TYPE,RONAME,1,1,COLNAME,BOVALU,BOVALU)
GO TO 700

C--- (3-RO ARGUMENT IN CALL IS NUMBER OF ENTRIES SUBMITTED FOR
C--- OUTPUT - IN THIS CASE ONLY ONE SO THE 6-TH AND 7-TH
C--- ARGUMENTS WILL BE IGNORED)

C--- BOUND TYPE WAS "LOWER".
C--- AVERAGE BDVALU (AGGREGATE BOUND VALUE) OVER NPR (NUMBER OF PERIODS
C--- IN AGGREGATION)
C--- OUTPUT BOUND CARD BY CALLING SUBROUTINE CARDOUT AND
C--- BRANCH TO "NEW NAME/TYP"E"

C---

BDVALU = BDVALU/NPR
CALL CARDOUT(TYPE,ROWNAME,1,1COLNAME,BDVALU,1COLNAME,BDVALU)
GO TO 700

C---
C--- BOUND TYPE WAS EITHER "FIXED" OR "UPPER".
C--- COMPARE NBOUNDS (NUMBER OF BOUNDS ENCOUNTERED) WITH NPR (NUMBER
C--- OF PERIODS IN AGGREGATION)
C--- IF EQUAL- TREAT SAME AS LOWER BOUND - BRANCH TO "TYPE WAS LOWER"

C---

690 IF (NBOUNDS.EQ.NPR) GO TO 685

C--- INCORRECT NUMBER OF BOUNDS ENCOUNTERED. (ASSUME TOO FEW)
C--- (SINCE DEFAULT UPPER BOUND IS INFINITY THE AVERAGE UPPER BOUND
C--- MUST BE INFINITY)

C--- PRINT WARNING
C--- IF UPPER BOUND DO NOT OUTPUT A BOUND CARD BUT
C--- BRANCH TO "NEW NAME/TYP"E"
C--- IF FIXED Bound CHANGE TYPE TO "LOWER" AND
C--- BRANCH TO "TYPE WAS LOWER"

C---

WRITE (6,958) TYPE,ROWNAME,1COLNAME,BDVALU,NPR,NBOUNDS
958 FORMAT (IH1V4A1,8A1,2X,BAI,2X,F12.6,'BOUNDS EXPECTED ',13,
'TOOS FEW BOUND ENCOIDERED',13,*,TOO FEW BOUNDS')
1 IF (TYPE(3).EQ.UP(2)) GO TO 700
TYPE(2) = LO(I)
TYPE(3) = LO(2)
GO TO 685

C--- NEW NAME/TYP ENCOUNTERED.
C--- IF BREAK CAUSED BY "ENDATA" CARD BRANCH TO "END SEGMENT"

C---

700 DO 710 I=I+1,4

C--- IF TYPE ON CURRENT CARD NOT EQUAL TO 'E','N','D','A'
C--- BRANCH TO "RESET OUTPUT BUFFERS"

C---

710 CONTINUE

GO TO 800
C--- RESET OUTPUT BUFFERS  -  SAVE NEWNAME AS COLNAME
C---  SAVE TYPIN AS TYPE
C---  SAVE VALUE AS BDVALU
C---  RESET NBOUNDS TO ONE

720 DO 730 I=1,8
730 CONTINUE
DO 735 I=1,4
735 CONTINUE
BDVALU = VALUE
NBOUNDS = 1

C--- IF INEWPR (INDEX OF OUTPUT PERIOD NUMBER) IS INVALID SET NPR TO ONE AND GO READ A NEW CARD
C---
C--- IF (INEPR.GT.0) GO TO 740
NPR = 1
GO TO 620

C--- VALID INEWPR.
C--- LOOKUP NPR (NUMBER OF PERIODS IN AGGREGATION) IN TABLE LISTIN
C--- (I-TH NUMBER IN LISTIN IS NUMBER OF PERIODS FROM INPUT MODEL TO BE AGGREGATED WHEN FORMING I-TH PERIOD OF OUTPUT MODEL)
740 NPR = LISTIN(INEWPR)

C--- GO READ A NEW BOUNDS CARD
C---
C--- GO TO 620

C---
C---
C---
C---
C---
C---
C--- Subroutine RENAME
898. C---
899. C---  - INPUT OLDNAME (ANY 8 CHARACTER NAME)
900. C---  - SELECTS LAST TWO NON-BLANK CHARACTERS OF NAME
901. C---  - CONVERTS THESE TO A (TWO-DIGIT) INTEGER NUMBER
902. C---  - RETURNS IF NOT A VALID INPUT PERIOD NUMBER
903. C---  - CONVERTS TO (TWO-CHARACTER) NEW PERIOD NUMBER
904. C---  - ACCORDING TO AGGREGATION SCHEME
905. C---  - SUBSTITUTES NEW PERIOD NUMBER FOR OLD AT END OF
906. C---  - NAME
907. C---  - RETURNS WITH NEWNAME (8 CHARACTER NAME WITH NEW
908. C---  - PERIOD NUMBER - IF VALID)
909. C---  - INEWPR (INTEGER EQUIVALENT OF NEW
910. C---  - PERIOD NUMBER)
911. C---  - NBLANK (NUMBER OF BLANKS AT END OF
912. C---  - NAME)
913. C---
914. C---  SUBROUTINE RENAME(OLDNAME,NEWNAME,INEWPR,NBLANK)
915. C---
916. C---  CHARACTER*1 OLDNAME(8),OLDPR(2),NEWNAME(8),NEWPR(2),BLANK
917. C---  CHARACTER*1 NAME(100,2)
918. C---  INTEGER INOUT(20)
919. C---  COMMON /BLOCK1/NPRIN,NPROUT,INOUT,PRNAME
920. C---
921. C---  COMMON BLOCK1 VARIABLES -
922. C---  (NPRIN - NUMBER OF PERIODS IN INPUT MODEL)
923. C---  (NPROUT - NUMBER OF PERIODS IN OUTPUT MODEL)
924. C---  (INOUT - LAST INPUT PERIOD NUMBER FOR EACH CORRESPONDING
925. C---  OUTPUT PERIOD NUMBER - I.E. AGGREGATION SCHEME)
926. C---  (PRNAME - TWO CHARACTER EQUIVALENTS FOR EACH POSSIBLE PERIOD
927. C---  NUMBER - I.E. ('0','9') TO ('9','9'))
928. C---
929. C---  INITIALIZE BLANK TO BLANK
930. C---  NBLANK TO ZERO
931. C---  INEWPR TO ZERO AND
932. C---  OLDPR TO ('0','0')
933. C---
934. C---  DATA BLANK/' '/
935. C---  NBLANK = 0
936. C---  INEWPR = 0
937. C---  OLDPR(1) = PRNAME(1,1)
938. C---  OLDPR(2) = PRNAME(1,2)
939. C---
940. C---  PROCESS OLDNAME CHARACTER BY CHARACTER BEGINNING WITH
941. C---  LAST CHARACTER AND INITIALLY SET NEWNAME EQUAL TO OLDNAME
942. C---
943. C---  DO 10 I=1,8
944. C---    L = 9 - I
945. C---    NEWNAME(L) = OLDNAME(L)
946. C---
947. C---  COMPUTE NBLANK (NUMBER OF BLANKS ON END OF OLDNAME)
948. C---  (ASSUME NO BLANKS OCCUR WITHIN THE BODY OF THE NAME)
949. C---
950.      IF (OLDNAME(L).EQ.BLANK) NBLANK = NBLANK + 1
951.      C---
952.      COMPUTE NONBLK (NUMBER OF NONBLANK CHARACTERS PROCESSED
953.      SO FAR)
954.      C---
955.      NONBLK = I - NBLANK
956.      C---
957.      PLACE LAST TWO NONBLANK CHARACTERS IN OLDPR
958.      C---
959.      IF (NONBLK.GT.2.OR.HOLBLK.EQ.0) GO TO 10
960.      INDEX = 3 - NONBLK
961.      OLDPR(INDEX) = OLDNAME(L)
962.      10      CONTINUE
963.      C---
964.      IF OLDNAME IS ALL BLANKS, RETURN
965.      C---
966.      IF (NBLANK.EQ.0) RETURN
967.      C---
968.      CONVERT OLDPR TO INTEGER EQUIVALENT BY CALLING SUBROUTINE CONVERT
969.      C---
970.      CALL CONVERT(OLDPR, IOLDPR)
971.      C---
972.      (IOLDPR = INTEGER EQUIVALENT OF OLDPR
973.      = EQUAIS ZERO IF NOT A VALID PERIOD NUMBER)
974.      C---
975.      IF OLDPR WAS NOT A VALID PERIOD NUMBER, RETURN
976.      C---
977.      IF (IOLDPR.EQ.0) RETURN
978.      C---
979.      COMPARE IOLDPR WITH INOUT (LIST OF ENDING PERIOD NUMBERS)
980.      C---
981.      30      DO 40 I=1,NPROUT
982.      C---
983.      IF (IOLDPR.GT.INOUT(I)) GO TO 40
984.      C---
985.      I IS NOW OUTPUT PERIOD NUMBER CORRESPONDING TO OLDPR.
986.      C---
987.      SAVE I AS INEWPR
988.      C---
989.      GET CHARACTER EQUIVALENT OF I FROM PRNAME AND
990.      C---
991.      SAVE IN NEWPR
992.      C---
993.      BRANCH TO "CHANGE PERIOD NUMBER"
994.      C---
995.      40      CONTINUE
996.      C---
997.      NO MATCH FOUND. (OLDPR MUST BE INVALID PERIOD NUMBER)
998.      C---
999.      RETURN (INEWPR WILL EQUAL ZERO)
1000.     C---
1001.     C--- CHANGE PERIOD NUMBER WITHIN NEWNAME TO NEW PERIOD NUMBER
1002.     C--- AND RETURN
NEWNAME(INDEX) = NEWPR(1)
NEWNAME(INDEX+1) = NEWPR(2)
RETURN
END

SUBROUTINE CONVERT

INPUT AB (ANY TWO CHARACTER COMBINATION)
- IF BOTH CHARACTERS ARE VALID DIGITS,
CONVERTS AB TO INTEGER EQUIVALENT, CALLED NUMBER
- OTHERWISE, SETS NUMBER = 0
RETURNS WITH NUMBER

SUBROUTINE CONVERT(AB,NUMBER)

CHARACTER*1 AB(2),DIGIT(10)
INTEGER N(2)

DATA DIGIT(),DIGIT(23,IT,DIDI(3,DIGIT(4)'0','1','2','3'/
DATA DIGIT(5),DIGIT(6),DIGIT(7),DIGIT(8)/'4','5','6','7'/
DATA DIGIT(9),DIGIT(10)/'8','9'/

INITIALIZE NUMBER TO ZERO
 NUMBER = 0

PROCESS INPUT CHARACTERS IN TURN (LOOP OVER I)
 DO 20 I=1,2
 N(I) = 0
20 CONTINUE

COMPARE CHARACTER WITH EACH OF TEN DIGITS (LOOP OVER J)
 DO 10 J=1,10
 IF (AB(I).NE.DIGIT(J)) GO TO 10
10 CONTINUE

I-TH CHARACTER EQUALS J-TH DIGIT.
SAVE AS N(I) AND GO ON TO NEXT CHARACTER
 N(I) = J-1
 GO TO 20
10 CONTINUE
1054. C--- I-TH CHARACTER IS NOT A VALID DIGIT.
1055. C--- RETURN (WITH NUMBER EQUAL TO ZERO)
1056. C---
1057. C--- RETURN
1058. 20 CONTINUE
1059. C--- N HOLDS DIGIT EQUIVALENTS OF AB CHARACTERS.
1060. C--- COMPUTE NUMBER (INTEGER EQUIVALENT OF AB) AND RETURN
1061. C--- NUMBER = 10^N(1) + N(2)
1062. C--- RETURN
1063. END
1064. C---
1065. C--- SUBROUTINE UPDATE
1066. C---
1067. C--- INPUT LASTiX (INDEX OF LAST ENTRY IN OUTPUT TABLES)
1068. C--- ROWNAME (ROW NAME OF ENTRY TO BE ADDED TO TABLES)
1069. C--- ROWVALU (ASSOCIATED AGGREGATE MPS MATRIX ENTRY)
1070. C--- PBLANK (NUMBER OF BLANKS AT END OF ROWNAME)
1071. C---
1072. C--- IF MATCHING ROW NAME IS FOUND IN TABLE NAMETAB THEN
1073. C--- ROWVALU IS ADDED TO CORRESPONDING ENTRY IN VALUTAB
1074. C--- IF NO MATCH IS FOUND NEW ENTRIES ARE SET UP IN NAMETAB
1075. C--- AND VALUTAB AND LASTiX IS INCREMENTED BY ONE
1076. C--- RETURNS WITH NEW VALUE OF LASTiX
1077. C---
1078. C--- SUBROUTINE UPDATE(LASTix, ROWnam, ROWVal, PBLanK, ARith, GEOM, COUNT, MINUSi)
1079. C---
1080. C--- COMMON BLOCK2 VARIABLES
1081. C--- (COLNAME - OUTPUT NAME OF AGGREGATED COLUMN)
1082. C--- (NAMETAB - LIST OF AGGREGATED ROW NAMES ENCOUNTERED FOR
1083. C--- THIS COLUMN)
1084. C--- (VALUTAB - CORRESPONDING LIST OF AGGREGATED MPS MATRIX ENTRIES)
1085. C--- (MAXENT - MAXIMUM NUMBER OF ENTRIES IN NAMETAB/VALUTAB)
1086. C--- SHOULD EQUAL DIMENSION)
1087. C---
1088. C--- IF NO PREVIOUS ENTRIES IN OUTPUT TABLES
1089. C--- BRANCH TO "NEW OUTPUT TABLE ENTRY"
1103. C---
1104. 10 IF (LASTIX.EQ.0) GO TO 50
1105. C---
1106. C--- COMPARE ROWNAME TO EACH ENTRY IN NAMETAB
1107. C--- (CONSIDER ONLY NONBLANK CHARACTERS IN ROWNAME)
1108. C---
1109. NONBLK = 8 - PBLANK
1110. DO 30 IX=1,LASTIX
1111. GO TO 20 L=1,NONBLK
1112. C---
1113. C--- IF NO MATCH, GO ON TO NEXT NAMETAB ENTRY
1114. C---
1115. 20 CONTINUE
1116. C---
1117. C--- MATCHING NAME FOUND IN NAMETAB.
1118. C--- ADD ROWVALU TO CORRESPONDING ENTRY IN VALUTAB
1119. C--- RESET ROWVALU TO ZERO AND RETURN
1120. C---
1121. C--- IF (ARITH) GO TO 22
1122. C--- IF (MINUSI) GO TO 24
1123. VALUTAB(IX)=VALUTAB(IX)+COUNT*(ROWVALU**(I/COUNT))
1124. C---
1125. GO TO 25
1126. VALUTAB(IX)=VALUTAB(IX)-(COUNT*(ROWVALU**(1/COUNT)))
1127. C---
1128. GO TO 23
1129. VALUTAB(IX)=VALUTAB(IX)+ROWVALU
1130. C---
1131. ROWVALU = 0.
1132. C---
1133. CONTINUE
1134. C---
1135. C--- NO MATCHING ROW NAME IN NAMETAB. (I.E. ROWNAME HAS NOT
1136. BEEN ENCOUNTERED BEFORE FOR THIS (OUTPUT) COLUMN NAME)
1137. C---
1138. C--- IF TABLES ARE NOT FULL, BRANCH TO "NEW OUTPUT TABLE ENTRY"
1139. C---
1140. 40 IF (LASTIX.LT.MAXENT) GO TO 50
1141. C---
1142. C--- OUTPUT TABLES ARE FULL.
1143. C--- PRINT WARNING AND RETURN
1144. C---
1145. C--- WRITE (6,952) MAXENT,COLNAME
1146. 952 FORMAT (1,H2,** NAMETAB/VALUTAB DIMENSION,' ,IS,
1147. 1 ' , EXTENDED FOR COLUMN ',A1)
1148. C--- (THE NUMBER OF SUCH WARNING MESSAGES WILL INDICATE THE
1149. EXTENT OF REDIMENSIONING REQUIRED - DON'T FORGET TO
1150. REDIMENSION IN SUBROUTINE COLOUT)
1151. C---
1152. RETURN
1153. C---
1154. C--- NEW OUTPUT TABLE ENTRY.
INCREMENT LASTIX (INDEX OF LAST ENTRY)

INSERT ROWNAME IN NAMETAB

INSERT ROWVALU IN VALUTAB

RESET ROWVALU (FOR SAFETY)

LASTIX = LASTIX + 1

DO 60 L=1,8

NAMETAB(LASTIX,L) = ROWNAME(L)

CONTINUE

IF (ARITH) GO TO 70

IF (MINUS) GO TO 65

VALUTAB(LASTIX) = COUNT*(ROWVALU**(1/COUNT))

COUNT=1

GO TO 80

65 VALUTAB(LASTIX) = (-1)*COUNT*(ROWVALU**(1/COUNT))

GO TO 66

70 VALUTAB(LASTIX) = ROWVALU

ROWVALU = 0.

IF (GEOM) GO TO 81

ARITH=TRUE

GO TO 82

ARITH=FALSE

RETURN

END

SUBROUTINE COLOUT

- INPUT LASTIX (INDEX OF LAST ENTRY IN OUTPUT TABLES)

- ASSUME VALID INDEX

- PROCESSES OUTPUT TABLES IN COMMON BLOCK2 TWO ENTRIES

- AT A TIME, SUBMITTING THEM TO SUBROUTINE CARDOUT FOR OUTPUT

- SHOULD EQUAL DIMENSION

CHARACTER*4 COLNAME(8), NAMETAB(100,8), NAME1(8), NAME2(8)

DIMENSION VALUTAB(100)

COMMON/BLOCK2/COLNAME, NAMETAB, VALUTAB, MAXENT

COMMON BLOCK2 VARIABLES -

(COLNAME - OUTPUT NAME OF AGGREGATED COLUMN)

(NAMETAB - LIST OF AGGREGATED ROW NAMES ENCOUNTERED FOR THIS COLUMN)

(VALUTAB - CORRESPONDING LIST OF AGGREGATED MPS MATRIX ENTRIES)

(MAXENT - MAXIMUM NUMBER ENTRIES IN NAMETAB/VALUTAB )
C--- INITIALIZE TYPE TO BLANKS
C--- DATA TYPE(1),TYPE(2),TYPE(3),TYPE(4), ' ', ' ', ' ', '/
C--- COMPUTE LEVEN (LARGEST EVEN NUMBER LESS OR EQUAL TO LASTIX) AND
C--- NEVEN (NUMBER OF EVEN NUMBERS LESS OR EQUAL TO LASTIX)
C---
1197. NEVEN = LASTIX/2
1198. LEVEN = NEVEN*2
1200. LOOP ONCE FOR EVERY TWO ENTRIES IN OUTPUT TABLES (LOOP OVER I)
1202. IF (NEVEN.EQ.0) GO TO 25
1203. DO 20 I=1,NEVEN
1204. INDEX = 2*I - 1
1205. MOVE NAMES FROM NAMETAB INTO NAME1 AND NAME2
1206. CALL STATEMENT WILL NOT ACCEPT AN IMPLIED DO LOOP
1208. DO 10 J=1,8
1209. NAME1(J) = NAMETAB(INDEX,J)
1210. NAME2(J) = NAMETAB(INDEX+1,J)
1212. 10 CONTINUE
1215. CALL CARDOU(TYPE,COLNAME,1,NAME2,VALUTAB(LASTIX))
1219. 20 CONTINUE
1221. IF (LASTIX.EQ.LEVEN) RETURN
1223. DO 30 J=1,0
1224. NAME1(J) = NAMETAB(LASTIX,J)
1225. NAME2(J) = NAME1(J)
1226. 30 CONTINUE
1228. CALL CARDOU(TYPE,COLNAME,1,NAME2,VALUTAB(LASTIX))
1230. (3-RD ARGUMENT IN CALL IS NUMBER OF ROW ENTRIES SUBMITTED FOR
1231. OUTPUT - IN THIS CASE ONLY ONE SO 6-TH AND 7-TH ARGUMENT
1232. WILL BE IGNORED)
1233. RETURN
1235. END
C---
1241. C--- SUBROUTINE CARDOUT
1242. C---
1243. C---  - INPUT TYPE (4 CHARACTER (BOUND) TYPE)
1244. C---  NAME1 (8 CHAR. COLUMN/RHS/BOUND NAME)
1245. C---  NAME2 (8 CHAR. NAME OF SECOND ENTRY)
1246. C---  NAME1 (8 CHAR. NAME OF FIRST ENTRY)
1247. C---  VALUE1 (FIRST ENTRY VALUE)
1248. C---  VALUE2 (SECOND ENTRY VALUE)
1249. C---
1250. C---  PRINTS WARNING IF ANY VALUE ENTRY IS ZERO
1251. C---  (UNLESS THIS IS A BOUND CARD)
1252. C---  - COUNTS NUMBER OF DIGITS TO LEFT OF DECIMAL
1253. C---  FOR EACH VALUE ENTRY
1254. C---  - SELECTS APPROPRIATE FORMAT STATEMENT
1255. C---  - WRITES OUT ONE CARD AND RETURNS
1256. C---
1257. C---
1258. C--- SUBROUTINE CARDOUT(TYPE,COLNAME,NENTRY,NAME1,VALUE1,NAME2,
1259.  VALUE2)
1260. C---
1261. C--- CHARACTER*1 TYPE(4),COLNAME(8),NAME1(8),NAME2(8),BLANK
1262. C---
1263. C--- INITIALIZE BLANK
1264. C---
1265. C--- THEN, IF INPUT TYPE FIELD NOT BLANK (IE. A BOUND CARD),
1266. C---
1267. C--- BYPASS TEST FOR ZERO ENTRY
1268. C---
1269. C---
1270. C--- INITIALIZE EPSILON (TOLERANCE FOR ZERO)
1271. C---
1272. C--- EPSILON = 0.000001
1273. C---
1274. C--- IF A SUBMITTED ENTRY IS WITHIN EPSILON OF ZERO, PRINT WARNING
1275. C---
1276. C---
1277. C--- IF (ABS(VALUE1).LT.EPSILON)
1278.  C---  WRITE (6,954) COLNAME,NAME1,VALUE1
1279. C---
1280. C---  IF (ABS(VALUE2).LT.EPSILON.AND.NENTRY.NE.1)
1281. C---  WRITE (6,954) COLNAME,NAME2,VALUE2
1282. C---  954 FORMAT (1H1,'COLUMN NAME ',8A1,' ROW NAME ',8A1,' VALUE ',
1283.  F12.6,' ** ZERO ENTRY')
1284. C---
1285. C--- BRANCH ON ABSOLUTE VALUE OF FIRST ENTRY
1286. C---
1287. C--- (LARGEST NUMBER ANTICIPATED IS 9,999,999,999)
1288. C---
1289. C--- 5 IF (ABS(VALUE1).LT.10000.) GO TO 10
1290. C---
1291. C--- 5 IF (ABS(VALUE1).LT.100000.) GO TO 70
1292. C---
C---
C FIRST ENTRY IS WITHIN 10,000. OF ZERO.
C---
C--- IF ONLY ENTRY, OUTPUT AND RETURN
C---
10 IF (NENTRY.NE.1) GO TO 20
19 WRITE (9,931) TYPE,COLONAME,NAME1,VALUE1
1500. RETURN

C---
C--- BRANCH ON ABSOLUTE VALUE OF SECOND ENTRY
C---
20 IF (ABS(VALUE2).LT.10000.) GO TO 30
205 IF (ABS(VALUE2).LT.100000.) GO TO 40
206 IF (ABS(VALUE2).LT.1000000.) GO TO 50
207. GO TO 60
208. C---
209. C--- SECOND ENTRY IS WITHIN 10,000 OF ZERO.
210. C--- OUTPUT AND RETURN
211. C---
30 WRITE (9,931) TYPE,COLONAME,NAME1,VALUE1,NAME2,VALUE2
213. RETURN

C---
C--- SECOND ENTRY IS BETWEEN 10,000 AND 100,000.
C---
39 WRITE (9,932) TYPE,COLONAME,NAME1,VALUE1,NAME2,VALUE2
219. RETURN

C---
C--- SECOND ENTRY IS BETWEEN 100,000 AND 1,000,000.
C---
50 WRITE (9,933) TYPE,COLONAME,NAME1,VALUE1,NAME2,VALUE2
225. RETURN

C---
C--- SECOND ENTRY IS GREATER OR EQUAL 1,000,000.
C---
60 WRITE (9,934) TYPE,COLONAME,NAME1,VALUE1,NAME2,VALUE2
231. RETURN

C---
C--- FIRST ENTRY IS BETWEEN 10,000 AND 100,000.
C---
70 IF (NENTRY.NE.1) GO TO 80
79 WRITE (9,935) TYPE,COLONAME,NAME1,VALUE1
239. RETURN

C---
C--- BRANCH ON ABSOLUTE VALUE OF SECOND ENTRY
1346. IF (ABS(VALUE2).LT.10000.) GO TO 90
1347. IF (ABS(VALUE2).LT.100000.) GO TO 100
1348. IF (ABS(VALUE2).LT.1000000.) GO TO 110
1349. GO TO 120
1350. C---
1351. C--- SECOND ENTRY IS WITHIN 10,000 OF ZERO.
1352. C--- OUTPUT AND RETURN
1353. C---
1354. 90 WRITE (9,935) TYPE, COLNAME, NAME1, VALUE1, NAME2, VALUE2
1355. RETURN
1356. C---
1357. C--- SECOND ENTRY IS BETWEEN 10,000 AND 100,000.
1358. C--- OUTPUT AND RETURN
1359. C---
1360. 100 WRITE (9,936) TYPE, COLNAME, NAME1, VALUE1, NAME2, VALUE2
1361. RETURN
1362. C---
1363. C--- SECOND ENTRY IS BETWEEN 100,000 AND 1,000,000.
1364. C--- OUTPUT AND RETURN
1365. C---
1366. 110 WRITE (9,937) TYPE, COLNAME, NAME1, VALUE1, NAME2, VALUE2
1367. RETURN
1368. C---
1369. C--- SECOND ENTRY IS GREATER OR EQUAL 1,000,000.
1370. C--- OUTPUT AND RETURN
1371. C---
1372. 120 WRITE (9,938) TYPE, COLNAME, NAME1, VALUE1, NAME2, VALUE2
1373. RETURN
1374. C---
1375. C---
1376. C---
1377. C FIRST ENTRY IS BETWEEN 100,000 AND 1,000,000.
1378. C---
1379. C--- IF ONLY ENTRY, OUTPUT AND RETURN
1380. C---
1381. C---
1382. 130 IF (NENTRY.NE.1) GO TO 140
1383. C---
1384. C---
1385. 140 WRITE (9,939) TYPE, COLNAME, NAME1, VALUE1
1386. RETURN
1387. C---
1388. C--- BRANCH ON ABSOLUTE VALUE OF SECOND ENTRY
1389. C---
1390. 150 IF (ABS(VALUE2).LT.10000.) GO TO 150
1391. IF (ABS(VALUE2).LT.100000.) GO TO 160
1392. IF (ABS(VALUE2).LT.1000000.) GO TO 170
1393. GO TO 150
1394. C---
1395. C--- SECOND ENTRY IS WITHIN 10,000 OF ZERO.
1396. C--- OUTPUT AND RETURN
1397. C---
1398. C---
1399. 150 WRITE (9,939) TYPE, COLNAME, NAME1, VALUE1, NAME2, VALUE2
1397. RETURN
1398. C--- SECOND ENTRY IS BETWEEN 10,000 AND 100,000.
1399. C--- OUTPUT AND RETURN
1400. C---
1401. 160 WRITE (9,940) TYPE,COLNAME,NAME1,VALUE1,NAME2,VALUE2
1402. RETURN
1403. C---
1404. C--- SECOND ENTRY IS BETWEEN 100,000 AND 1,000,000.
1405. C--- OUTPUT AND RETURN
1406. C---
1407. 170 WRITE (9,941) TYPE,COLNAME,NAME1,VALUE1,NAME2,VALUE2
1408. RETURN
1409. C---
1410. C--- SECOND ENTRY IS GREATER OR EQUAL 1,000,000.
1411. C--- OUTPUT AND RETURN
1412. C---
1413. 180 WRITE (9,942) TYPE,COLNAME,NAME1,VALUE1,NAME2,VALUE2
1414. RETURN
1415. C---
1416. C---
1417. C---
1418. C---
1419. C--- FIRST ENTRY IS GREATER OR EQUAL 1,000,000.
1420. C---
1421. C--- IF ONLY ENTRY, OUTPUT AND RETURN
1422. C---
1423. 190 IF (ENTRY.NE.1) GO TO 200
1424. 190 IF (ENTRY.NE.1) GO TO 200
1425. C---
1426. 190 WRITE (9,944) TYPE,COLNAME,NAME1,VALUE1
1427. RETURN
1428. C---
1429. C--- BRANCH ON ABSOLUTE VALUE OF SECOND ENTRY
1430. C---
1431. 200 IF (ABS(VALUE2).LT.10000.) GO TO 210
1432. IF (ABS(VALUE2).LT.100000.) GO TO 220
1433. IF (ABS(VALUE2).LT.1000000.) GO TO 230
1434. GO TO 240
1435. C---
1436. C--- SECOND ENTRY IS WITHIN 10,000 OF ZERO.
1437. C--- OUTPUT AND RETURN
1438. C---
1439. 210 WRITE (9,944) TYPE,COLNAME,NAME1,VALUE1,NAME2,VALUE2
1440. RETURN
1441. C---
1442. C--- SECOND ENTRY IS BETWEEN 10,000 AND 100,000.
1443. C--- OUTPUT AND RETURN
1444. C---
1445. 220 WRITE (9,945) TYPE,COLNAME,NAME1,VALUE1,NAME2,VALUE2
1446. RETURN
1447. C---
1448. C--- SECOND ENTRY IS BETWEEN 100,000 AND 1,000,000.
1449. C--- OUTPUT AND RETURN
write (9,946) type, colname, name1, value1, name2, value2
return

C---

C--- second entry is greater or equal 1,000,000.
C--- output and return
C---
write (9,946) type, colname, name1, value1, name2, value2
return
C---
C--- format statements
C---
format (4a1,8a1,2x,8a1,2x,f12.6,3x,8a1,2x,f12.6)
format (4a1,8a1,2x,8a1,2x,f12.6,3x,8a1,2x,f12.5)
format (4a1,8a1,2x,8a1,2x,f12.6,3x,8a1,2x,f12.4)
format (4a1,8a1,2x,8a1,2x,f12.6,3x,8a1,2x,f12.3)
format (4a1,8a1,2x,8a1,2x,f12.5,3x,8a1,2x,f12.6)
format (4a1,8a1,2x,8a1,2x,f12.5,3x,8a1,2x,f12.5)
format (4a1,8a1,2x,8a1,2x,f12.5,3x,8a1,2x,f12.4)
format (4a1,8a1,2x,8a1,2x,f12.5,3x,8a1,2x,f12.3)
format (4a1,8a1,2x,8a1,2x,f12.4,3x,8a1,2x,f12.6)
format (4a1,8a1,2x,8a1,2x,f12.4,3x,8a1,2x,f12.5)
format (4a1,8a1,2x,8a1,2x,f12.4,3x,8a1,2x,f12.4)
format (4a1,8a1,2x,8a1,2x,f12.4,3x,8a1,2x,f12.3)
format (4a1,8a1,2x,8a1,2x,f12.3,3x,8a1,2x,f12.6)
format (4a1,8a1,2x,8a1,2x,f12.3,3x,8a1,2x,f12.5)
format (4a1,8a1,2x,8a1,2x,f12.3,3x,8a1,2x,f12.4)
format (4a1,8a1,2x,8a1,2x,f12.3,3x,8a1,2x,f12.3)
format (4a1,8a1,2x,8a1,2x,f12.3,3x,8a1,2x,f12.2)
format (4a1,8a1,2x,8a1,2x,f12.3,3x,8a1,2x,f12.1)
format (4a1,8a1,2x,8a1,2x,f12.3,3x,8a1,2x,f12.0)

end

C---

C---

C---

C---

C---

C---

C---

$DATA
1 2 3 1 1
$STOP
/*
*/