EUROPEAN SPACE AGENCY
DIRECTORATE OF OPERATIONS
FLIGHT CONTROL SYSTEMS DEPARTMENT

ESATAN: AN EVALUATION

Technical Note

DOPS-SST-TN-0215-SIM

ISSUE: 1 REV: 0

4 March 1994

This page is left blank intentionally for double-sided printing

DOCUMENT APPROVAL

Prepared by	Organization	Signature	Date
Miguel Á. Carreira Perpiñán	ESOC/SIM		

Approved by	Organization	Signature	Date
Juan E. Miró	ESOC/SIM		

Authorised by	Organization	Signature	Date
J. J. Gujer	H/ESOC/SIM		

DOCUMENT STATUS SHEET

DOCUMENT STATUS SHEET			
1. DOCUMENT TITLE: ESATAN: An Overview			
2. DOCUMENT REFERENCE NUMBER: DOPS-SST-TN-0215-SIM			
3. ISSUE	4. REVISION	5. DATE	6. REASON FOR CHANGE
1	0	4/3/94	First Issue

DOCUMENT CHANGE RECORD

DOCUMENT CHANGE RECORD	DCR NO	
	DATE	
	ORIGINATOR	
	APPROVED BY	

1. DOCUMENT TITLE
2. DOCUMENT REFERENCE NUMBER
3. DOCUMENT ISSUE/REVISION NUMBER:

4. PAGE	5. PARAGRAPH	6. REASON FOR CHANGE

ABSTRACT

This paper[1] presents ESATAN, the new ESA thermal network analysis program. A description of its main features and structure is given, as well as an outline of the mathematical algorithms involved. One simple example of use is given. Finally, the program is briefly discussed from the user's point of view and its performance evaluated.

Questions, comments, suggestions, death threats etc. are welcome.

Miguel Á. Carreira Perpiñán ☺
Darmstadt, March 1994

[1] This report has been elaborated under the sponsorship of the Dirección General de Investigación Científica y Técnica (Spain).

CONTENTS

Page

DOCUMENT APPROVAL i

DOCUMENT STATUS SHEET ii

DOCUMENT CHANGE RECORD iii

ABSTRACT iv

CONTENTS v

GLOSSARY OF TERMS vii

DISTRIBUTION LIST viii

LIST OF FIGURES ix

1 INTRODUCTION 1

2 STRUCTURE OF ESATAN 2

3 THE "LUMPED PARAMETER" METHOD 4
3.1 The General Lumped Parameter Method Equations 4

4 ESATAN'S MAIN FEATURES 6

5 GENERAL STRUCTURE OF AN INPUT DECK 8

6 A WORKED EXAMPLE 9
6.1 Input Deck for Model LBAR 11
6.2 Checked Listing for Model LBAR 12
6.3 Reference Listing for Model LBAR 13
6.4 Output Listing for Model LBAR 17

7 OUR INSTALLATION OF ESATAN 22

8 USER ORIENTATION EVALUATION 23

9 PERFORMANCE EVALUATION 25
9.1 Introduction 25
9.2 Preprocessing Performance 25
9.2.1 Fit model 25
9.2.2 Fit formula 26
9.3 Running performance 26
9.3.1 Generic input deck chosen 27
9.3.2 Fit model 28
9.3.3 Fit formula 29
9.3.4 When is accurate the former formula? 29
9.3.5 Performance of ESATAN for transient-state simulations 30
9.3.6 Is ESATAN able to perform a real-time simulation? 32

9.4 The SOHO Satellite Input Deck 33
9.5 ESATAN v5.5 vs ESATAN v6.3 36

LIST OF REFERENCES 37

APPENDIX A: Regression Fits A-1

APPENDIX B: Experimental Data B-1

APPENDIX C: Fit Formulae for the Performance Evaluation C-1
Preprocessing time C-1
Running time for our skeleton input deck C-1
Real-time capability for a simplified transient state simulation scheme . . . C-1

GLOSSARY OF TERMS

CGM: Computer Graphics Metafile. A widespread vector graphics format.

DTIMEI: input timestep for transient solution routines (ESATAN control constant).

DTIMEU: timestep length for a transient solution routine (ESATAN control constant).

ESATAN: ESA Thermal ANalyser. Program reviewed in this note.

GKS: Graphical Kernel System. A standard set of graphics primitives.

Input deck, source deck: text file containing a specification for a thermal mathematical model.

LOG file: file produced by the VMS operating system. It contains statistical data about the execution of a task (elapsed time, CPU time, page faults...).

Mortran: ESATAN-proprietary superset of Fortran 77. It includes structured constructs (SELECT...CASE, WHILE...ENDWHILE and REPEAT...UNTIL), subscripted values, etc.

Steady state: state that a (thermal) system reaches when time goes to infinity. It does not always exist.

Transient: evolution of a (thermal) system during a given time interval.

NLOOP: maximum number of iterations of a solution routine (ESATAN control constant).

OUTINT: output interval (ESATAN control constant).

RELXCA: convergence threshold of a solution routine (ESATAN control constant).

SOHO: SOlar and Heliospheric Observatory. Satellite for which a validated ESATAN source deck was available.

TIMEND: time at end of transient (ESATAN control constant).

LIST OF FIGURES

Figure 1: The Basic ESATAN System 3
Figure 2: Bar model 9
Figure 3: Derived thermal network for the bar model 9
Figure 4: Skeleton source deck used for the measures 30
Figure 5: Transient-state simulation with ESATAN 33

1 INTRODUCTION

ESATAN (*ESA Thermal ANalyser*) is, as stated in the User's Manual, a software package for the prediction of temperature distributions in engineering components and systems using the thermal network analysis technique.

It is intended to replace ESA's former thermal analysis program, SINDA. It has been developed by the Engineering Research Centre of GEC Alsthom, Whetstone, using Fortran 77 to ANSI Standard X-3.9 1978 and it should run on any computer supporting such a compiler with only minor changes dependant on the operating system (file naming rules...).

The development and simulation of a thermal model with ESATAN involves three stages:

- Creation by the user of an input deck, that is, a text file containing, in **Mortran**[2] language, a specification for the model (definition of the model topology, associated tables of information regarding material properties, variation of model parameters with time, etc.) and instructions for the solution procedure.

- Preprocessing this input deck by the ESATAN preprocessor. This means checking whether the model is both syntactically and semantically consistent and then, if it is, creating a file in ESATAN's binary notation for this model as well as a Fortran program. These files are internal to ESATAN and the user should not be concerned about them. All errors occurred during the preprocess are logged onto a listing file.

- Solving the model. The former Fortran source program is compiled and linked with the system libraries plus ESATAN's own library. The resulting executable file is automatically run and its output sent to a text file which is the desired solution[3]. The EXE file may be run as often as desired.

The sequence of operations is very similar to that of editing, compiling and running a program.

Deriving a network model from a specified physical system is, of course, the part which requires most work for the user.

---

[2] Mortran is an ESATAN-proprietary superset of Fortran 77. It includes structured constructs (SELECT...CASE, WHILE...ENDWHILE and REPEAT...UNTIL), subscripted values, etc.

[3] Basically, a solution consists of a snapshot with the predicted temperatures throughout the system. Such snapshots can be taken at intervals, thus giving a view of the evolution of the system (its transient state). When time goes to infinity the system may reach a steady state; this steady-state solution can also be calculated by ESATAN.

2 STRUCTURE OF ESATAN

ESATAN itself consists of three major components:

■ The **Syntax Checker**: a fast, preliminary program to check the correctness of the syntax of a source deck. It is intended to reduce the edit-preprocess-edit cycle, as the preprocessor takes a considerable time to run for moderate to large sized input decks (☞ see some sample preprocessing times under the epigraph *Performance Evaluation*). It won't detect every error, only the simpler ones.

■ The **Preprocessor**: reads an input deck, checks it for consistency and translates it into a Fortran program.

■ The **Library**: a collection of object modules which can be called from the instructions section of a user input deck.

This structure is depicted in fig. 1.

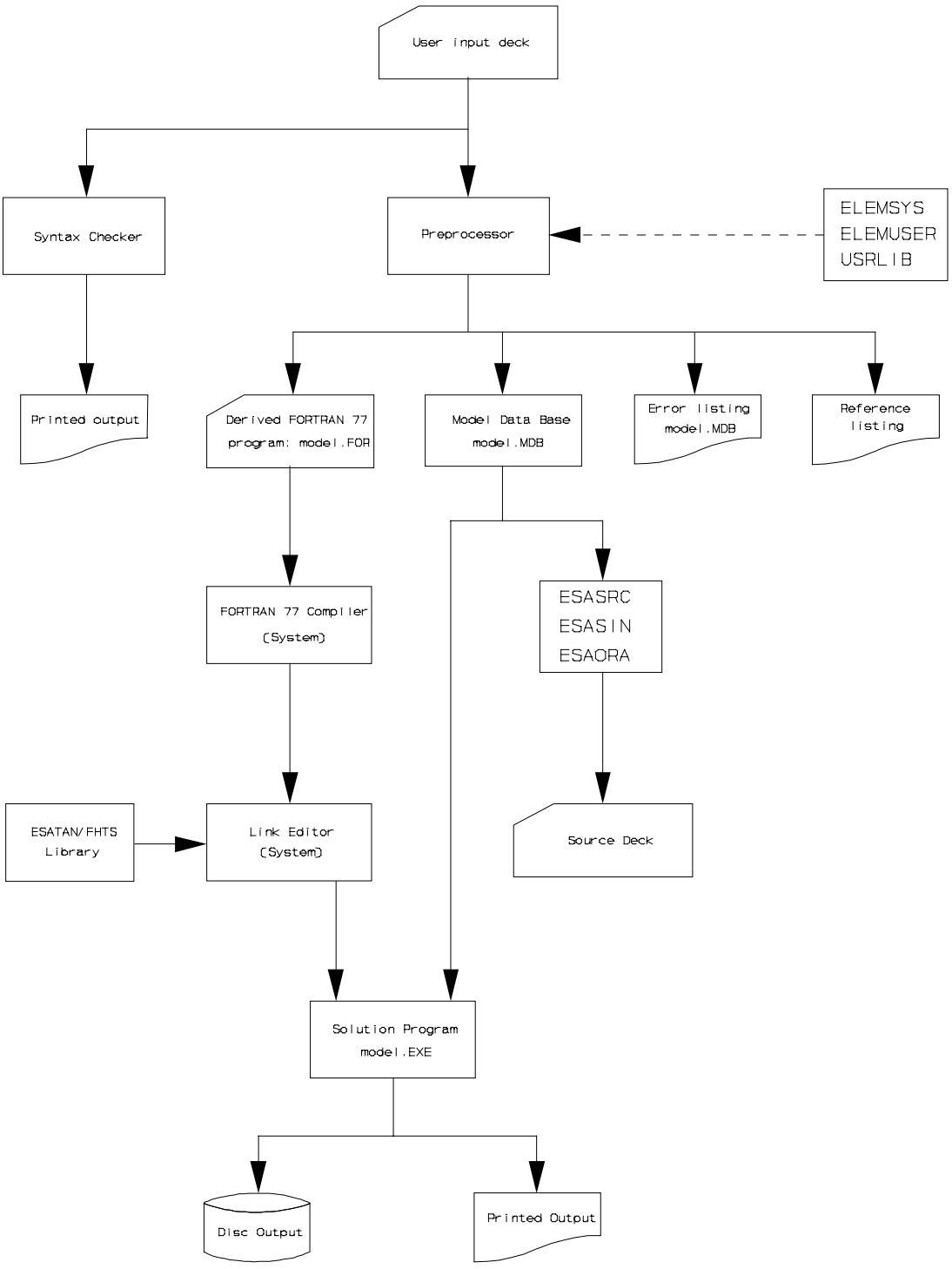


Figure 2: The Basic ESATAN System

3 THE "LUMPED PARAMETER" METHOD

The key idea in this method is to model a continuous medium as a discrete network of **nodes** (representing the capacitance of a particular section of the system) linked by **conductors** (representing its conductance). All the physical parameters of the thermal system, which are spatially distributed in it, are "lumped" in a set of quantities and assigned to each node or conductor. This means that a node or conductor is considered as a black box defined by its temperature, capacitance, etc. considered as constant throughout this node or conductor. This introduces a discretisation error which will be small if the network is fine. In the numerical algorithm, governing differential equations are approximated by means of first-order finite differences.

There is a thermal/electric analogy in which temperature equates to voltage and heat flow to current flow. The equivalent for a thermal network is an electric circuit.

3.1 The General Lumped Parameter Method Equations

This section is intended to give to the interested reader a rough, summarised idea of the underlying mathematics of thermal modelling.

Given a network with n nodes $i = N_1, ..., N_n$, each having temperature T_i, heat capacity C_i and containing a heat source Q_i; and being K_{ij} the linear conductance (conduction, convection and other linear processes such as evaporation or condensation) and R_{ij} the radiative exchange constant between nodes N_i and N_j; then, applying a heat balance to node i and taking into account the heat diffusion equation[4] and the Stefan-Boltzmann law[5], one can see that, for $i = 1, ..., n$:

---

[4] The heat diffusion differential equation, which gives the variation of temperature with time, is

$$\frac{\partial T}{\partial t} = \alpha \nabla^2 T, \qquad \alpha = \frac{\kappa}{\rho c}$$

or, in the form of Fourier's law

$$\boldsymbol{q} = -\kappa \nabla T = \frac{\boldsymbol{Q}}{A}$$

where q is the heat flow (W m^2), $\boldsymbol{Q}$ is the rate of heat transfer (W), A is the area for heat transfer, and α, κ, c and ρ are the thermal diffusivity (m^2 s^{-1}), thermal conductivity (W/m-°C), specific heat (J/kg-°C) and density (kg m^{-3}) of the material, respectively. ∇ is the nabla operator and ∇^2 is the laplacian operator.

[5] For a grey body, the rate of radiated heat is

$$Q = \sigma \alpha \varepsilon \mathcal{F} A T^4$$

where σ is the Stefan-Boltzmann constant ($\approx 5.6697 \times 10^{-8}$ W/m^2-°K^4), α is the absorptivity (-), ε is the emissivity (-), $\mathcal{F}$ is the view factor (-), A is the area of radiation (m^2) and T its temperature (°C).

$$C_i \frac{dT_i}{dt} = \sum_{i \neq j} K_{ij}(T_j - T_i) + \sum_{i \neq j} R_{ij}(T_j^4 - T_i^4) + Q_i \qquad (transient\ case)$$

$$0 = \sum_{i \neq j} K_{ij}(T_j - T_i) + \sum_{i \neq j} R_{ij}(T_j^4 - T_i^4) + Q_i \qquad (steady\ state\ case)$$

In the transient case there are n coupled first-order differential equations which may be integrated numerically and in the steady state case it reduces to a non-linear system of n equations with n unknowns T_i (the system will be linear[6] if there is no radiation present).

One needs to know the values for the thermal conductivity κ, the specific heat c, the density ρ and the geometric parameters (thicknesses, distances etc.) and then derive from them the values for the lumped parameters C_{ij}, K_{ij} and R_{ij}. A wide variety of methods may be used to achieve this, depending on the geometry and other factors.

For more information the reader is referred to the Engineering Manual of ESATAN. Examples are given there for:

- Discretising plates and cylinders;
- Calculating the associated conductances;
- Calculating view factors;

as well as pointers to the specialised literature, which provides with formulae for more complex geometries (handbooks, etc.).

[6] If the conductivity κ varies with the temperature (which is the normal case), then the system is always non-linear.

4 ESATAN'S MAIN FEATURES

■ Allowed heat transfer types are conductive, convective and radiative, any of which can be non-linear with respect to temperature or time.

■ Physical parameters like specific heat, heat capacitance, emissivity area etc. can vary arbitrarily with temperature, time, position or other variables.

■ A number of routines are provided for:

- Both steady-state and transient solutions (there are several different methods to choose from).

- Mathematical routines: Lagrangian interpolation, approximate integration, matrix operations (including transposition, inversion and determinant evaluation), polynomial substitution, least squares fit, etc.

- Error reporting.

- Output formatting: facilities are provided to print values for the temperature, capacitance, heat sources (albedo, internal, etc.), heat flow, infrared emissivity and many other variables, as well as tables, averages, sums, maxima, balances etc.

- Plotting: X-Y arrays or one variable against time. ESATAN can produce *Computer Graphics Metafiles* (CGM) thanks to the use of the *Graphical Kernel System* (GKS).

- Others: array sort, scaling...

■ Useful physical constants like **g** (acceleration due to gravity) or σ (Stefan-Boltzmann) are built-in.

■ The units for all quantities are those of the *Système International d'Unités* (SI) except for temperature, which is measured by default in Celsius degrees (Kelvin, Fahrenheit or Rankine scales are possible too).

■ Models may be built in a hierarchical way: each model can in turn have several submodels, resulting in a tree structure with up to 99 levels.

■ Submodels can be combined either by node merging or by inter-model conductance links.

■ Implicit model definition by the inclusion of previously defined submodels.

■ Batch editor permitting implicit model definitions to be modified.

■ The user can control the solution process (convergence, accuracy, etc.) by specifying other values than the default for certain parameters (eg. number of iterations, convergence criterion...).

■ Extensions to ESATAN:

- FHTS (*Fluid Heat Transport System*): models fluid networks giving a solution for pressure, mass-flow rate and temperature/enthalpy. A library with solution routines and standard elements (pumps, condensers, etc.) is supplied, as well as a graphical front end. Boiling, condensation, melting, solidification, gas-solid transitions and other phase-change phenomena can be modelled.

- ESARAD: separate program to generate 3D thin-shell models.

- ABLAT: ablation modelling facility[7].

■ Some commented example input decks are provided, as well as a Training Manual.

■ Also includes an ORACLE database interface which allows storing an ESATAN model in an ORACLE database (ESAORA).

■ There are utilities to translate an ESATAN input deck to/from a SINDA model (ESASIN) and to obtain an input deck from a model database (ESASRC).

[7] Under development.

5 GENERAL STRUCTURE OF AN INPUT DECK

A source deck is made up of **blocks**. There are two types of blocks: *operations* and *data* blocks. Data blocks are used to define the topology of the network, as well as to give values to physical constants and properties. Operations blocks are used to give instructions on how the solution will be reached and what outputs will be printed.

The most important data blocks are[8]:

$NODES: identify nodes and give initial temperature for them and, optionally, capacitance, heat sources, area, absorptivity and emissivity. Each node can belong to one of three types: diffusion, boundary (it represents a boundary condition, i.e. its temperature is constant) and inactive (they aren't taken into account in the solution).

$CONDUCTORS[9]: define links between nodes and give the value for its conductance. There are four different types, namely linear (conduction through a solid or convection between a solid and a fluid), radiative, fluidic (one-way mass-flow) and radiative with view-factors.

$CONSTANTS: user-defined variables and control constants like *DTIMEI* (input time step for transient solution), *RELXCA* (convergence criterion), *TIMEND* (time at end of transient solution), *NLOOP* (maximum no. of iterations), *OUTINT* (output interval), among others. Arrays of values can also be defined.

The most important operations blocks are:

$EXECUTION: all solution and user-defined[10] routines must be called from this block.

$OUTPUTS: contains operations to be performed at each output interval (*OUTINT*) for a transient solution or at the end of a steady state solution.

A hash (#) anywhere (except inside string constants) means that the rest of the line is a comment. Lines following the **$MODEL** line are also handled as comments until the start of the next **$KEYWORD** line. Needless to say, one should properly comment source decks — neither too much nor too little.

---

[8] Each data or operations block starts with a **$KEYWORD** line and ends with the **$KEYWORD** line of the following block.

[9] Data in the **$NODES** and **$CONDUCTORS** blocks may be defined as literal values, variables, variable expressions or user defined functions.

[10] In a **$SUBROUTINES** block.

6 A WORKED EXAMPLE

The following example is a model of a bar insulated along its length, radiating to deep space at both ends and containing a constant heat source at one end. All material properties are constant. The bar is represented by five diffusion nodes, the two end nodes being half the volume of the three internal nodes. Deep space is represented by boundary node B10. Figure 2 shows the bar and figure 3 the derived thermal network.

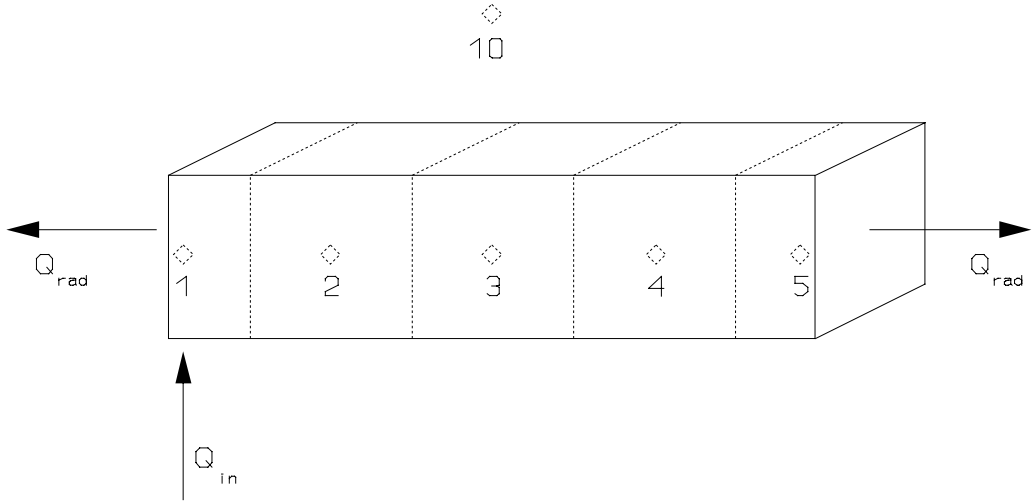


Figure 3: Bar model

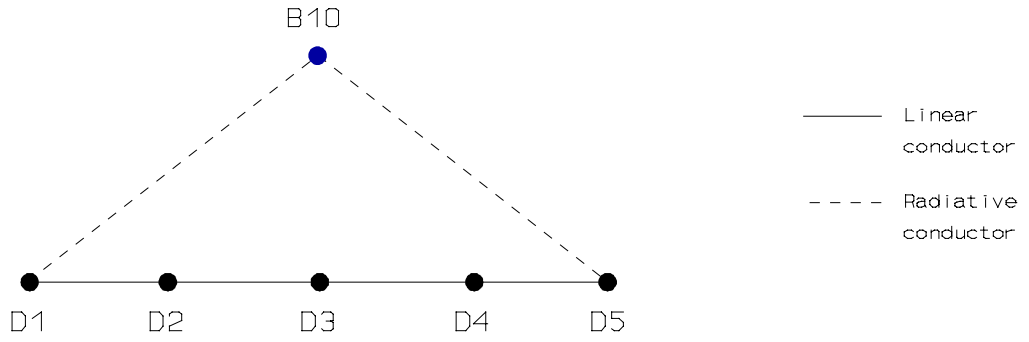


Figure 4: Derived thermal network for the bar model

The reference listing and solution output files are given following the listing of the input deck. General features of the reference listing are its standardised format, model level number and line numbering, and data block information summary. The output file contains information at the top of each file including date, time, page number and solution control constant values.

☞ Note that, in order to get some snapshots of the evolution of this system, a transient solution routine is used (*SLFRWD*) with time limit *TIMEND* = 100 seconds and output interval *OUTINT* = 30 seconds. In this case a steady state solution routine might not converge because there is a constant heat source which keeps temperature growing up with time continuously throughout the bar and this may not be balanced by radiation heat loss through both ends, thus preventing the system from reaching equilibrium[11].

[11] Running ESATAN with the *SOLVIT* solution routine shows that in this case a steady state actually exists. The following is *SOLVIT*'s output:

```
1
 EUROPEAN  SPACE  AGENCY  THERMAL  ANALYSIS  NETWORK                          (VERSION 5.5)                        PAGE  1
  7 MARCH 1994                                                 10:00:07                                                   LBAR

  Ejemplillo migueluno de prueba  --  09:28:00, 07/03/94



  TIMEN =      0.00      MODULE SOLVIT      LOOPCT =   100      ENBALA =    0.0366      ENBALR = 0.0005


 TABLE OUTPUT WITH ZENTS = 'L,T'
 FOR NODES OF ZLABEL = ' '

 ==========================================================================================================================

+LBAR


       NODE    LABEL                              T

          1    Bar with end source            865.82
          2    Bar                            862.77
          3    Bar                            859.72
          4    Bar                            856.67
          5    Bar end                        853.61
         10                                  -273.15
```

6.1 Input Deck for Model LBAR

```
$MODEL LBAR

        Ejemplillo migueluno de prueba  --  09:28:00, 07/03/94

        LINEAR BAR MODEL
#
$NODES
#
D1 = 'Bar with end source',
T = 26.67, C = 38.925, QI = 70.0;
D3 = 'Bar', T = 26.67, C = 77.85;       # C = MASS * SPECIFIC HEAT CAPACITY
D2 = 'Bar', T = 26.67, C = 77.85;       # NODE NOS. NEED NOT BE IN ORDER
D4 = 'Bar', T = 26.67, C = 77.85;
D5 = 'Bar end', T = 26.67, C = 38.925;
B10, T = -273.15;
#
$CONDUCTORS
# Linear
GL(1, 2) = 11.25;
GL(2, 3) = 11.25;
GL(3, 4) = 11.25;
GL(4, 5) = 11.25;
# Radiative
GR(1, 10) = 3.75E-4;
GR(5, 10) = 3.75E-4;
#
$CONSTANTS
#
$CONTROL
#
TIMEND = 100.0;         # TIME AT END OF TRANSIENT
OUTINT = 30.0;          # OUTPUT INTERVAL
RELXCA = 0.01;          # CONVERGENCE CRITERION
NLOOP = 100;            # MAXIMUM NO. OF ITERATIONS
#
# OPERATIONS BLOCKS NEXT
#
$EXECUTION
#
        HEADER = '@(#)case01.d1.11      10:22:32,92/10/01'
        CALL SLFRWD
#
$OUTPUTS
#
        CALL PRNDTB(' ', 'L,T', CURRENT)
#
$ENDMODEL LBAR
```

6.2 Checked Listing for Model LBAR

```
    @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
    @                            @
    @  ESATAN SYNTAX CHECKER     @
    @                            @
    @  @SYN@ - syntax error      @
    @  @SEM@ - semantic error    @
    @  @WAR@ - warning           @
    @                            @
    @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

$MODEL LBAR

        Ejemplillo migueluno de prueba  --  09:28:00, 07/03/94

        LINEAR BAR MODEL
#
$NODES
#
D1 = 'Bar with end source',
T = 26.67, C = 38.925, QI = 70.0;
D3 = 'Bar', T = 26.67, C = 77.85;       # C = MASS * SPECIFIC HEAT CAPACITY
D2 = 'Bar', T = 26.67, C = 77.85;       # NODE NOS. NEED NOT BE IN ORDER
D4 = 'Bar', T = 26.67, C = 77.85;
D5 = 'Bar end', T = 26.67, C = 38.925;
B10, T = -273.15;
#
$CONDUCTORS
# Linear
GL(1, 2) = 11.25;
GL(2, 3) = 11.25;
GL(3, 4) = 11.25;
GL(4, 5) = 11.25;
# Radiative
GR(1, 10) = 3.75E-4;
GR(5, 10) = 3.75E-4;
#
$CONSTANTS
#
$CONTROL
#
TIMEND = 100.0;         # TIME AT END OF TRANSIENT
OUTINT = 30.0;          # OUTPUT INTERVAL
RELXCA = 0.01;          # CONVERGENCE CRITERION
NLOOP = 100;            # MAXIMUM NO. OF ITERATIONS
#
# OPERATIONS BLOCKS NEXT
#
$EXECUTION
#
        HEADER = '@(#)case01.d1.11      10:22:32,92/10/01'
        CALL SLFRWD
#
$OUTPUTS
#
        CALL PRNDTB(' ', 'L,T', CURRENT)
#
$ENDMODEL

 LBAR


  @@@@ ERROR MESSAGE SUMMARY  @@@@

       0 syntax   errors in input file

       0 semantic errors in input file

       0 warnings        in input file
```

6.3 Reference Listing for Model LBAR

```
                              EUROPEAN  SPACE  AGENCY  THERMAL  ANALYSIS  NETWORK

                EEEEEEEEEEE    SSSSSSSSS     AAAAAAAAAA    TTTTTTTTTTT    AAAAAAAAAA    NN        NN
                EEEEEEEEEEE   SSSSSSSSSSS   AAAAAAAAAAAA   TTTTTTTTTTT   AAAAAAAAAAAA   NNN       NN
                EE            SS        SS  AA        AA       TT        AA        AA   NNNN      NN
                EE            SS            AA        AA       TT        AA        AA   NN NN     NN
                EE            SSS           AA        AA       TT        AA        AA   NN  NN    NN
                EEEEEEEE       SSSSSSSSS    AAAAAAAAAAAA       TT        AAAAAAAAAAAA   NN   NN   NN
                EEEEEEEE         SSSSSSSSS  AAAAAAAAAAAA       TT        AAAAAAAAAAAA   NN    NN  NN
                EE                     SSS  AA        AA       TT        AA        AA   NN     NN NN
                EE                      SS  AA        AA       TT        AA        AA   NN       NNNN
                EE            SS        SS  AA        AA       TT        AA        AA   NN        NNN
                EEEEEEEEEEE   SSSSSSSSSSS   AA        AA       TT        AA        AA   NN         NN
                EEEEEEEEEEE    SSSSSSSSS    AA        AA       TT        AA        AA   NN         NN

                              EUROPEAN  SPACE  AGENCY  THERMAL  ANALYSIS  NETWORK


DESIGNED AND DEVELOPED:  ENGINEERING RESEARCH CENTRE                SUPPORT: USERS REQUIRING SUPPORT FOR USE OF THIS
                         WHETSTONE                                           PROGRAM SHOULD CONTACT:-
                         LEICESTER
                         U.K.                                  CENTRAL SUPPORT: C J KIRTLEY, ERC,  ...  "  ...  "  ...  "  ...
                                                                                N J STOCK,   ERC,  (44)-533-750750  EXT 3521.
        EUNET ADDRESS:   esatan@gec-erc.co.uk
                                                               LOCAL  SUPPORT: ADD LOCAL SUPPORT NAME HERE  O is letter o

INSTALLATION          :  ADD INSTALLATION NAME HERE   0 is zero

VERSION               :  5.5

ISSUE DATE            :  APRIL 1992

================================================================================================================================

MODEL NAME     :  LBAR

DATA BASE FILES:  LBAR.MDB   LBAR.FOR

RUN DATE       :  19 MAY 1993         11:20:14

EDIT NO.       :     0

ERRORS         :  NONE

================================================================================================================================
```

```
ESATAN REFERENCE LISTING FOR MODEL LBAR                    19 MAY 1993                                    PAGE 1

 0     1 $MODEL LBAR , REAL , LIST , STANDARD
 0     2
 0     3         Ejemplillo migueluno de prueba  --  09:28:00, 07/03/94
 0     4
 0     5         LINEAR BAR MODEL
 0     6 #


 0     7 $LOCALS
 0     8    $INTEGER
 0     9    $REAL
 0    10    $CHARACTER

          ************************************************************************
          $LBAR                                                             $NODES
          ************************************************************************

 0    11 $NODES
 0    12 #THERMAL
 0    13 D1 = 'Bar with end source' ,           T = 26.67 ,         C = 38.925 ,
          QI = 70.0 ;         D2 = 'Bar' ,      T = 26.67 ,         C = 77.85 ;
 0    15 D3 = 'Bar' ,         T = 26.67 ,       C = 77.85 ;
 0    16 D4 = 'Bar' ,         T = 26.67 ,       C = 77.85 ;
 0    17 D5 = 'Bar end' ,     T = 26.67 ,       C = 38.925 ;
 0    18 B10 ,                T =  - 273.15 ;
 0    19 #FLUID

              DATA SUMMARY FOR NODES
                     5*DIFFUSION
                     1*BOUNDARY

          ************************************************************************
          $LBAR                                                       $CONDUCTORS
          ************************************************************************

 0    20 $CONDUCTORS
 0    21 #   LINEAR
 0    22 GL(1 , 2) = 11.25 ; GL(2 , 3) = 11.25 ; GL(3 , 4) = 11.25 ; GL(4 , 5) = 11.25 ;
 0    26 #   RADIATIVE
 0    27 GR(1 , 10) = 3.75E-4 ;                 GR(5 , 10) = 3.75E-4 ;
 0    29 #   FLUIDIC
 0    30 #   VIEW FACTOR
 0    31 #   LINEAR INTER-MODEL
 0    32 #   RADIATIVE INTER-MODEL
 0    33 #   FLUIDIC INTER-MODEL
 0    34 #   MASS
 0    35 #   FITTING LOSS
 0    36 #   MASS INTER-MODEL
 0    37 #   FITTING INTER-MODEL
```

```
            DATA SUMMARY FOR CONDUCTORS
                  4*   LINEAR
                  2*   RADIATIVE

         ************************************************************************
         $LBAR                                                      $CONSTANTS
         ************************************************************************

 0    38 $CONSTANTS
 0    39 #
 0    40 #
 0    41    $INTEGER
 0    42    $REAL
 0    43    $CHARACTER
 0    44    $CONTROL
 0    45  TIMEND =  100.0   ; OUTINT =  30.0    ; RELXCA =  0.01    ; NLOOP  =  100     ;

            DATA SUMMARY FOR CONSTANTS
                  4*  $CONTROL
 0    49 $ARRAYS
 0    50    $INTEGER
 0    51    $REAL
 0    52    $CHARACTER
 0    53    $TABLE
 0    54 $SUBROUTINES
 0    55 $INITIAL
 0    56 $VARIABLES1
 0    57 $VARIABLES2

         ************************************************************************
         $LBAR                                                     $EXECUTION
         ************************************************************************

 0    58 $EXECUTION
 0    59 #
 0    60         HEADER = '@(#)case01.d1.11      10:22:32,92/10/01'
 0    61         CALL SLFRWD
 0    62 #

         ************************************************************************
         $LBAR                                                      $OUTPUTS
         ************************************************************************

 0    63 $OUTPUTS
 0    64 #
 0    65         CALL PRNDTB(' ', 'L,T', CURRENT)
 0    66 #
 0    67 $ENDMODEL LBAR

================================================================================
```

ESATAN REFERENCE LISTING FOR MODEL LBAR 19 MAY 1993 PAGE 3

HIERARCHY OF MODELS INCLUDING VIRTUALS(V).

LEVEL	MODEL NAME
0	LBAR

========== END OF ESATAN REFERENCE LISTING FOR MODEL LBAR ==========

6.4 Output Listing for Model LBAR

```
EUROPEAN  SPACE  AGENCY  THERMAL  ANALYSIS  NETWORK                          (VERSION 5.5)                      PAGE     1
19 MAY 1993                                                  11:22:15                                                        LBAR

@(#)case01.d1.11      10:22:32,92/10/01



 TIMEN =      0.00      MODULE SLFRWD      DTIMEU =    3.2868

CSGMIN =    3.4598  AT NODE 1 IN SUB-MODEL                                                LBAR


TABLE OUTPUT WITH ZENTS = 'L,T'
FOR NODES OF ZLABEL = ' '

=================================================================================================================================

LBAR

      NODE    LABEL                              T

         1    Bar with end source             26.67
         2    Bar                             26.67
         3    Bar                             26.67
         4    Bar                             26.67
         5    Bar end                         26.67
        10                                  -273.15
```

EUROPEAN SPACE AGENCY THERMAL ANALYSIS NETWORK (VERSION 5.5) PAGE 2
19 MAY 1993 11:22:15 LBAR

@(#)case01.d1.11 10:22:32,92/10/01

TIMEN = 30.00 MODULE SLFRWD DTIMEU = 1.8527

CSGMIN = 3.4598 AT NODE 1 IN SUB-MODEL LBAR

TABLE OUTPUT WITH ZENTS = 'L,T'
FOR NODES OF ZLABEL = ' '

LBAR

NODE	LABEL	T
1	Bar with end source	41.25
2	Bar	35.91
3	Bar	32.22
4	Bar	30.08
5	Bar end	29.37
10		-273.15

EUROPEAN SPACE AGENCY THERMAL ANALYSIS NETWORK (VERSION 5.5) PAGE 3
19 MAY 1993 11:22:15 LBAR

@(#)case01.d1.11 10:22:32,92/10/01

TIMEN = 60.00 MODULE SLFRWD DTIMEU = 1.8528

CSGMIN = 3.4598 AT NODE 1 IN SUB-MODEL LBAR

TABLE OUTPUT WITH ZENTS = 'L,T'
FOR NODES OF ZLABEL = ' '

LBAR

NODE	LABEL	T
1	Bar with end source	48.22
2	Bar	42.80
3	Bar	38.92
4	Bar	36.60
5	Bar end	35.81
10		-273.15

```
EUROPEAN  SPACE  AGENCY  THERMAL  ANALYSIS  NETWORK                          (VERSION 5.5)                      PAGE     4
19 MAY 1993                                                  11:22:15                                                      LBAR

@(#)case01.d1.11      10:22:32,92/10/01



 TIMEN =     90.00      MODULE SLFRWD      DTIMEU =    1.8528

CSGMIN =    3.4598  AT NODE 1 IN SUB-MODEL                                                  LBAR


TABLE OUTPUT WITH ZENTS = 'L,T'
FOR NODES OF ZLABEL = ' '

=================================================================================================================================
LBAR

      NODE    LABEL                              T

         1    Bar with end source              54.94
         2    Bar                              49.51
         3    Bar                              45.63
         4    Bar                              43.29
         5    Bar end                          42.50
        10                                   -273.15
```

EUROPEAN SPACE AGENCY THERMAL ANALYSIS NETWORK (VERSION 5.5) PAGE 5
19 MAY 1993 11:22:15 LBAR

@(#)case01.d1.11 10:22:32,92/10/01

TIMEN = 100.00 MODULE SLFRWD DTIMEU = 1.7132

CSGMIN = 3.4598 AT NODE 1 IN SUB-MODEL LBAR

TABLE OUTPUT WITH ZENTS = 'L,T'
FOR NODES OF ZLABEL = ' '

LBAR

NODE	LABEL	T
1	Bar with end source	57.17
2	Bar	51.74
3	Bar	47.86
4	Bar	45.52
5	Bar end	44.73
10		-273.15

7 OUR INSTALLATION OF ESATAN

We are using ESATAN version 5.5[12] for the VAX and it is installed in our server, **SIMDS1**. Every user connected to it can execute the program. It is necessary to have Motif running under DECwindows in order to use the *FHTS Graphical Front End*. For the rest of possible actions, a text terminal is enough.

Presently, ESATAN is licensed for only one CPU (the server), so it won't run when started from a different VAXstation.

However, our installation is not complete: some modules, including ESARAD, ABLAT, ESAORA and the interfaces with window-systems are missing.

[12] Now we have version 6.3 as well.

8 USER ORIENTATION EVALUATION

■ ESATAN offers a spartan human interface, but you don't need much more anyway. The program is started by calling the command file ESATAN.COM, which in turn will call the corresponding executable image(s). The main menu is:

```
                                    ESATAN

WHICH OPTION DO YOU REQUIRE?
     SYNTAX CHECKER (C)
     PREPROCESS (P)
     SOLVE (S)
     BATCH (B)
     ESATAN SOURCE DUMP (D)
     ESATAN TO SINDA TRANSLATOR (E)
     SINDA TO ESATAN TRANSLATOR (T)
     FHTS GRAPHICAL FRONT END (G)
     QUIT (Q)
OPTION = :
```

Figure 4: ESATAN main menu

After selecting the desired option you are asked to give the appropriate file/model names.

The **S** option (*Solve*) will compile and link the FORTRAN program which resulted from the preprocess of the input deck, and furthermore it will run it thus getting the desired solution.

You can preprocess, solve or both in batch mode by selecting the **B** option (*Batch*).

The **D** option allows to get a source deck from a model database. The **E** and **T** options are useful for conversions between ESATAN and SINDA formats, although occasionally some manual corrections will be unavoidable, because they are not fully compatible.

■ Errors that occurred when using the menus are poorly described and don't even appear in the User's Manual. In general, it provides little information about how to start the program, choose options, etc. although it is quite straightforward. It would also be nice to have a printout of the initial screen (like above). The **G** option (*FHTS Graphical*

Front End) isn't documented at all.

■ The fact that ESATAN can presently be only run from the server could overload it when run from several terminals at once, especially if one takes into account that preprocessing a moderate-sized input deck is quite time-consuming (☞ see the *Performance Evaluation* epigraph).

■ It takes little time to learn the preprocessor syntax, but designing a network for a given system requires some knowledge of thermal systems (heat transfer foundations, main equations) and even of numerical analysis if a full profit is to be obtained from the library functions.

■ Basic FORTRAN programming ability is necessary.

■ For all purposes, the **documentation** about a model should include not only the source deck, but also schematics of the associated network, showing the relation between nodes and conductors with the physical system and explaining how the significative values (constants, capacitances, etc.) were obtained (or, at least, pointing to references: thermal handbooks, etc.), as well as a justification of the chosen derived network. That is of paramount importance in big systems but should not be overlooked in the smaller ones. A glossary would be more than welcome, specially when lots of acronyms are employed.

■ Creating an input deck with a text editor is a tough task and error-prone. A graphical interface is a must.

■ ESATAN allows the user to define local constants —visible only to the preprocessor— in the **$LOCALS** data block, but with a restricted naming: *ILn* (integer), *RLn* (real) and *ZLn* (character), where *n* is an integer. For the sake of clarity, it would be desirable to have more freedom when choosing constant names. Otherwise it is very common to end up with expressions like this one[13]:

```
          GL(1,408)=1./((1./((20.0*RL701)+(0.05*8.)))+
(2./((4.*RL314*RL106*RL153)/(LOG((RL701+A408)/(2.*RL701))))));
```

[13] Taken from the input deck prepared for the SOHO satellite.

9 PERFORMANCE EVALUATION

9.1 Introduction

A performance study of ESATAN is presented. The reader who is only interested in the result and not in how it was reached can jump directly to appendix III.

Here the two computational phases required for carrying out a simulation with ESATAN, namely **preprocessing** an input deck and **running** the resulting executable file, are discussed in terms of performance. The CPU consumption for a number of cases is presented and then an approximate general formula derived from these data (using a least-squares fit) is given.

The numerical measures for the preprocessing and execution time were obtained by preprocessing the input deck in batch mode (option B in ESATAN's main menu) and submitting the resulting executable file to the VAX system, respectively. In both cases the elapsed CPU time (that is, the CPU time used for preprocessing or running, without the time due to other users) can be found by inspecting the corresponding LOG file[14].

All the time values are give in seconds unless otherwise noted.

The actual system in which this performance evaluation was accomplished is our server, SIMDS1, a VAX 3800 rated with about 8 VUPs or some 8.4 MIPS[15].

9.2 Preprocessing Performance

9.2.1 Fit model

It is advisable to have an idea at least of the order of magnitude of the time that a preprocessing pass will take. Unfortunately the examples that ESATAN provides with are all small-sized but hopefully the fit given will still produce acceptable estimates for large-sized models (more than 500 nodes).

---

[14] Apart from other things, the LOG file gives: elapsed time (wall-clock time), CPU time and number of page faults. The VMS operating system uses virtual memory and page faults occur when a reference virtual address does not reside in RAM but on disk. The process of fetching some disk blocs (depending on the replacing strategy used) takes much longer than a RAM access. This introduces a major variability source in the time measured, as it is usually not possible to know *a priori* the average access time for a virtual memory address (it depends on the amount of memory installed, how many processes are running...).

[15] Indeed, while this document was being completed our server was upgraded from a VAX 3800 rated with *around* 8 VUPs to a VAX 4100 rated with 24 VUPs (approximately 25.2 MIPS); the architecture —and thus the instruction set— are similar, so one can expect a uniform performance improvement. I carried out some tests by running exclusively either ESATAN itself (the preprocessor) or executable files created using ESATAN and the resulting relative performance turned out to be about 2.6 for the VAX 4100. The only time measures made with the new server are those belonging to the SOHO satellite input deck and they have been multiplied by the factor 2.6, so that a comparison can be done.

The input deck is supposed to consist of *definition statements* (such as the definition of nodes and conductors) and *non-definition statements* (the rest: comments, executable statements, block headers etc.). In general, it will cost more to preprocess a definition statement than a non-definition one because the latter will be normally copied to the output FORTRAN file with almost no modifications (executable statements) or even discarded (comments). It seems thus reasonable to assume that the total preprocessing time will be proportional to the size (in bytes) of each class of statements.

The variables t, s and d will designate the preprocessing time (seconds), total size of the input deck (bytes) and size of the declarative part (bytes), respectively.

☞ The reader should bear in mind that this fit is very rough and is based on weakly justified assumptions; for instance, one obvious source of variation is the loop definition of a set of nodes or links (a couple of sentences can expand to many lines, each of which defines a single node or link, thus taking much more computer time).

9.2.2 Fit formula

The experimental measures are given in appendix II (table II.1).

On the triads (s,d,t) a linear bi-dimensional fit[16] was performed with the result

$$t^*(s,d) = As + Bd + C$$

$$A = 4.3\cdot10^{-4} \text{ s/byte} \quad B = 2.4\cdot10^{-4} \text{ s/byte} \quad C = 4.43 \text{ s}$$

where t^* is the fitted value of t given the values of d and t. The values of t are tabulated too for comparison with the experimental ones. The mean relative error[17] is 24% while having a maximum of 61%.

9.3 Running performance

The total CPU time spent in a run for a given model depends on a number of parameters: the number of nodes, the solution routine used, the number of iterations employed globally (steady state) or in each timestep (transient), the number of timesteps (transient state), the output interval, the contents of the **$VARIABLES1** and **$VARIABLES2** sections... It is by far too complicated to take into account all these factors. Besides, some of them are *a priori* unknown, like the number of iterations required. It is therefore necessary to restrict oneself to a very simplified input deck.

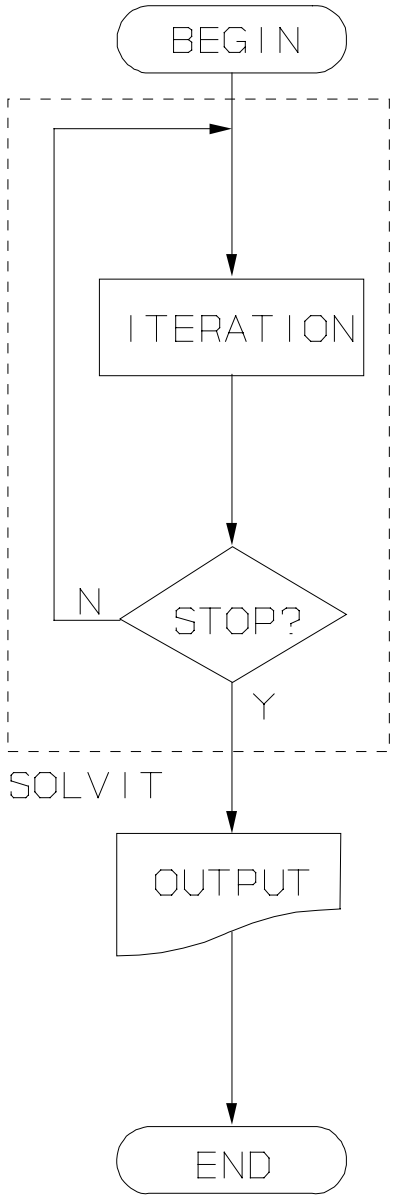
9.3.1 Generic input deck chosen

As shown in figure 4, the skeleton input deck chosen consists of a call to the

[16] See appendix I.

[17] See appendix I.

SOLVIT steady-state solution routine and a call to the *PRNDTB* (*Print Data Block*) output routine at the end of the simulation. There is no restriction on the number of nodes or conductors of the model.




```
     ....

     ....

     ....

(Definition of the network nodes

and conductors and other data)

     ....

     ....

     ....

$CONTROL

     RELXCA = 0;

     NLOOP = 100;

# 100 iterations will be carried out

$EXECUTION

     CALL SOLVIT

$OUTPUT

     CALL PRNDTB(' ', 'T ',CURRENT)

# Prints the number and temperature

# of each node
```

Figure 5: Skeleton source deck used for the measures

9.3.2 Fit model

The time required to simulate one of these skeleton models is equal to the time spent in the *SOLVIT* call plus the time consumed in initialising and printing the results:

1. Time spent in initialisation and output: it can be considered proportional to n because it is nothing more than printing one line per node plus a constant number of

lines. The load time of the program is almost constant, as the biggest part are the libraries (system, ESATAN...) which are linked with every model, regardless of its number of nodes. However, adding a linear dependency on *c* proves to improve considerably the goodness of the fit.

2. Time spent by *SOLVIT*: as shown, the *SOLVIT* routine consists basically of a loop which is repeated *i* times (the number of iterations). *i* can be easily controlled by setting the *RELXCA* control variable (convergence criterion: minimum difference in absolute value between consecutive temperatures for each node) to a very small value or just 0 (so that a lot of iterations will probably be necessary) and the setting the *NLOOP* control variable (maximum number of iterations allowed) to the actual number of iterations desired.

The time spent in each iteration, t_i, can be assumed constant for *n*, *c* fixed. Then the complexity in time of the *SOLVIT* routine will be linear in *i*.

So far the complexity in time for the skeleton model has the form:

$$t^*(n,c,i) = t_i(n,c)\,i + t_o(n,c)$$

For t_o we proposed above a linear dependency on *n* and *c*:

$$t_o(n,c) = \varepsilon n + \zeta c + \eta$$

This hypothesis was contrasted with experimental data[18] and the following values found for a linear-2D regression fit: $\varepsilon = 2.6\cdot10^{-3}$ s, $\zeta = 1.2\ -10^{-4}$ s, $\eta = 0.93$ s, with $\varepsilon = 5\%$ and $\varepsilon_{max} = 12\%$.

Now, we propose the following dependency of t_i (time per iteration) on *n* and *c*:

$$t_i(c,n) = \alpha c + \beta n^2 + \gamma n + \delta$$

That is, linear in *c* and quadratic in *n*. Both theoretical considerations and experimentation lead to this decision. Each iteration must deal with *n*×*n* matrices which are mostly composed of zeroes (they are **sparse matrices**). A conductor between nodes (*i*,*j*) will make elements (*i*,*j*) and (*j*,*i*) different from zero in a certain matrix; otherwise they will be zero, which is the case for most pairs (*i*,*j*) as not every node will be connected to all the remaining nodes in a real system. ESATAN seems to use a sparse matrix algorithm for some operations (that is, dependant linearly on the number of non-zero elements: in other words, *c*) and a normal one for others (dependant quadratically on *n*). Thus the expression above for t_i.

A regression fit was performed on it[19] with the data given in table II.2[20]. The

[18] Shown in appendix II.

[19] See appendix I.

[20] See appendix II.

results are[21]: $\alpha = 10^{-4}$ s, $\beta = 4 \cdot 10^{-7}$ s, $\gamma = 7.5 \cdot 10^{-5}$ s, $\delta = 0$ s with error bounds $\varepsilon = 7\%$ and $\varepsilon_{max} = 29\%$.

9.3.3 Fit formula

The time needed to run a skeleton model can be approximated as:

$$t(n, c, i) = (\alpha c + \beta n^2 + \gamma n + \delta) i + \varepsilon n + \zeta c + \eta$$

where $\alpha = 10^{-4}$, $\beta = 4 \cdot 10^{-7}$, $\gamma = 7.5 \cdot 10^{-5}$, $\delta = 0$, $\varepsilon = 2.6 \cdot 10^{-3}$, $\zeta = 1.2 \cdot 10^{-4}$, $\eta = 0.93$ (all the parameters are in seconds).

The error bounds for *t* are: $\varepsilon = 5\%$, $\varepsilon_{max} = 16\%$.

9.3.4 When is the former formula accurate?

For small or medium thermal models, the fit will produce good predictions as long as the models are kept "simple." It will underestimate the running time with, for example:

- Models that present high non-linearities, which oblige ESATAN to make interpolations on different magnitudes as a function of the temperature.

- Use of certain features which are computationally very expensive: *VCHP* (Variable Conductance Heat Pipe Control), *VFAC* (computation of radiative conductances for a view factor submodel), etc.

- ...

As the biggest model that the author could have access to has only 208 nodes, it is unknown whether the formula will still give proper values for big models (more then 500 nodes).

Nevertheless, it is hoped that it will be an acceptable estimate of at least the order of magnitude of the running time.

9.3.5 Performance of ESATAN for transient-state simulations

Essentially, what ESATAN does to dynamically calculate the nodal temperatures is to update them periodically. The time interval between updates, called **timestep**, must be small enough so that the required accuracy is attained. The whole simplified process is depicted in figure 5.

The timestep length, *DTIMEU*, is computed internally by ESATAN as

---

[21] The model *NLBAR* was discarded because it is a clear outlier. This is due to the high non-linearities that it contains: 85% of the nodes and 50% of the conductors interpolate material properties such as heat capacity or conductivity as a non-linear function of temperature and this takes more computer time when compared with the rest of the models used, which are quite linear.

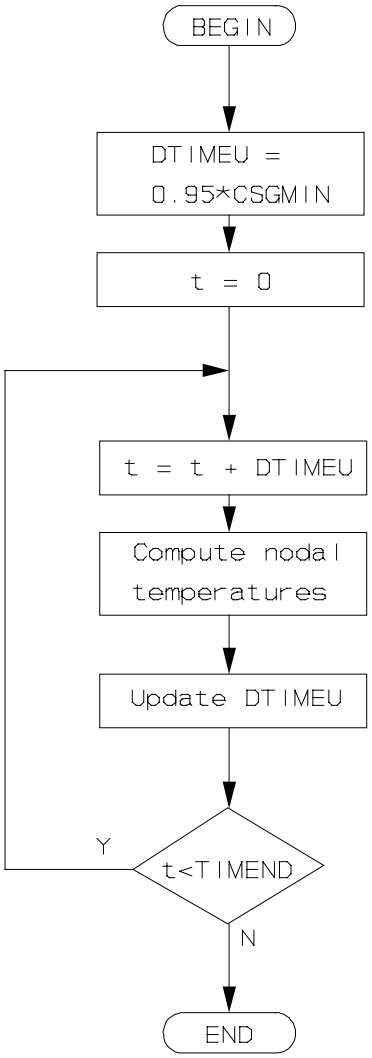


Figure 6: Transient-state simulation with ESATAN

$$DTIMEU = 0.95 * CSGMIN$$

$$CSGMIN = \min_{i=1,\ldots,n} \left\{ \frac{C_i}{\sum_{j \neq i} GL_{ij}} \right\}$$

where C_i is the heat capacitance of node i and GL_{ij} is the heat conductance of the link between nodes i and j. The expression

$$\frac{C_i}{\sum_{j \neq i} GL_{ij}}$$

has units of time and has the same physical meaning as $1/\tau = RC$ in a resistor-capacitor circuit: it is the time constant of a "thermal circuit." It has got a different value for each node and then one must take the smallest one to satisfy all them (thus the *min* of *CSGMIN*). The 0.95 factor gives some security slack.

The timestep length does not remain constant during the simulation because the C_i and GL_{ij} can vary with time. Therefore, it must be recalculated at the end of each timestep.

ESATAN provides with five different solution routines for the transient state. They differ in how they obtain the nodal temperatures for a new timestep (using forward differences, the Crank-Nicholson method, a predictor-corrector method...). Most of these methods operate by solving an $n \times n$ system of equations iteratively, that is, the same approach used for the steady-state solution routines. Then, knowing the number of iterations employed in each timestep one can find the total number of iterations and roughly approximate the running time with the fit formula given above (as n and c are already known).

The problem, of course, is: how does one know a priori the number of iterations? The answer is: it is almost impossible, because it depends on the mathematical nature of the matrices involved —as well as on the desired degree of accuracy (*RELXCA*, *NLOOP*).

The best thing one can do is to consider a constant timestep of 0.95 * *CSGMIN* as it is at the beginning (this relies on the fact that the material properties relevant for heat transfer won't usually change too much with temperature, which is a good assumption) and take an "average" number of iterations per timestep, $\bar{I}$. Then, the following expression will yield the grand total of iterations for the whole simulation:

$$i = \frac{TIMEND}{0.95 \times CSGMIN} \times \overline{I}$$

where the number of timesteps is

$$\frac{TIMEND}{0.95 \times CSGMIN}$$

(*TIMEND* is the time at the end of the transient state).

Alternatively, one can just leave unknown the number of iterations. Then we have the equation of a line (t parameterised on i).

The following table lists the predicted times[22] for some selected sample models. The values for the total number of iterations, i, were known in advance.

Model	n	c	i	*CSGMIN*	*TIMEND*	t	t^*
HEATER	168	1161	150	2677.961	8640	23	22
EVAP2	99	153	9	2000	10000	1.3	1.5
TESTCASE	11	27	312	0.0165	2	1.9	2.1
LBAR	6	6	1520	3.4598	100	3.4	2.6
TGMODAL	58	91	1498	6.7592	2500	27	23
TGTRAN	58	91	1496	6.7592	2500	26	23

One can see that the predictions are still good for the transient-state solution routines.

As an example, the values for the number of iterations and *CSGMIN* are tabulated below for the *LBAR* example. The subroutine used is *SLFWBK* (Crank-Nicholson —implicit forward-backward differencing). The initial value for *CSGMIN* occurs at node 1 and is around 3.45982 seconds. Notice how it changes slightly as the simulation goes on; in a more realistic case the changes will be greater.

[22] All of which (*CSGMIN*, *TIMEND*, t, t^*) are given in seconds.

Simulation time	Iterations	*CSGMIN*	Simulation time	Iterations	*CSGMIN*
3.286799907684326	18	3.459823738505052	55.87559843063354	100	3.459786378090168
6.573599815368652	17	3.459816189975047	59.16239833831787	16	3.459784873533585
9.860399723052979	100	3.459812237456282	62.44919824600220	15	3.459783369259751
13.14719963073730	19	3.459809259578331	65.73599815368652	100	3.459781863575965
16.43399953842163	18	3.459806757934513	69.02279806137085	16	3.459780355193491
19.72079944610596	18	3.459804549275543	72.30959796905518	100	3.459778843130432
23.00759935379028	17	3.459802540259411	75.59639787673950	17	3.459777326637939
26.29439926147461	19	3.459800674094908	78.88319778442383	16	3.459775805144148
29.58119916915894	18	3.459798912398994	82.16999769210815	16	3.459774278211593
32.86799907684326	17	3.459797227647376	85.45679759979248	100	3.459772745504870
36.15479898452759	100	3.459795599485074	88.74359750747681	100	3.459771206766071
39.44159889221191	18	3.459794012630688	92.03039741516113	15	3.459769661796132
42.72839879989624	100	3.459792455530702	95.31719732284546	100	3.459768110440671
46.01519870758057	100	3.459790919422188	98.60399723052979	17	3.459766552579233
49.30199861526489	100	3.459789397649417	101.8907971382141	13	3.459766552579233
52.58879852294922	100	3.459787885153972			

9.3.6 Is ESATAN able to perform a real-time simulation?

Let us consider the previous simplified transient state simulation scheme, that is, without outputs —just recomputing the nodal temperatures in each timestep. The CPU time required for a single timestep will be approximately $t_i \cdot \bar{I}$, being $\bar{I}$ the "average" number of iterations employed in it[23]. $t_i(n,c)$ is the time per iteration, as defined some pages before. Then, the quotient

$$\frac{t_i \times \bar{I}}{DTIMEU}$$

will give us the ratio between computer time and real time. The relation[24]

[23] For many observed cases it is 1 iteration per timestep.

[24] The "less than" operators (≤) should really mean "approximately less than" because of the simplifications introduced, but unhappily WordPerfect 5.2 —as many other things— lacks this symbol.

$$\frac{t_i \times \overline{I}}{DTIMEU} \leq 1 \quad \Leftrightarrow \quad t_i \times \overline{I} \leq DTIMEU$$

gives us the condition that ESATAN must satisfy in order to be "real time." It depends on n, c, $\bar{I}$, *DTIMEU* and the processor speed, which is 8.4 MIPS for our analysis[25]. The reader can verify that ESATAN runs in real time for all the former examples. For a bigger model with n = 500 nodes, c = 3000 conductors (typical for a satellite), $\bar{I}$ = 1 iteration per timestep and timesteps of 1 second (= *DTIMEU*), it will still support real time (the real-time ratio is 44%), but for an even bigger model with n = 800, c = 8000 it will break down (real time ratio of 112%). In this case, a more powerful processor should be used to ensure a real-time simulation.

9.4 The SOHO Satellite Input Deck

The author was given the opportunity to test ESATAN with the SOHO[26] Satellite Thermal Input Deck (Phase C/D 2, Service Module - SVM). This input deck is representative of a "real world" thermal model. It has been prepared by the spanish company Construcciones Aeronáuticas, S. A. (CASA) for use at ESTEC with ESATAN.

The entire input deck is divided into the following files, which once preprocessed build up the final model:

- *I100.DAT, I200.DAT, I300.DAT, I400.DAT*: these files contain the sub-models *I1, I2, I3* and *I4* respectively, which in turn correspond to the +Y+Z, -Y+Z, -Y-Z, +Y-Z quarters of the internal enclosure of the satellite in a reference system in which the main platform lies on the YZ plane.

- *E5CS00.DAT, E5HS00.DAT*: contain the sub-model for the external enclosure for two test cases, cold (*E5C*) and hot (*E5H*), respectively. In the cold case the total disipation power is 1441 watts while in the hot case it is set to 1353 watts; the difference is due to different power values in the heaters as well as putting to work selected sets of them.

- *ST3400.DAT*: contains the main model, which consists of *I1, I2, I3, I4* and exactly one of *E5C* and *E5H* (plus the inter-model conductors and some few new nodes and conductors that are added in *ST3400.DAT*).

---

[25] One can take t_i (and, in general, the rest of the time values) as inversely proportional to the processor speed. Then, the former expression generalised for a computer with a speed of *MIPS* turns into

$$\frac{t_i \times \overline{I}}{DTIMEU} \times \frac{8.4}{MIPS} \leq 1$$

[26] The name SOHO stands for "Solar and Heliospheric Observatory." It should be launched by the ATLAS rocket in mid-1995 and it will be placed on a halo orbit around the first Lagrangian point of the Sun-Earth system, L_1, which lies in the line Sun-Earth at 1.5 million kilometres from the Earth. Hence it never goes through the Earth shadow and it suffers no eclipses.

Basically, the **$EXECUTION** block of this file first sets the initial state (temperatures, heat flows etc.); then computes the steady state for the cold case (that is, *E5C* is enabled and *E5H* disabled); beginning from this point, the steady state for the hot case is computed (with *E5C* disabled and *E5H* enabled). The ESATAN library routine used to calculate the steady state is *SOLVIT*.

The following table summarises the size of the SOHO thermal mathematical model:

File name	File size (Kbytes)	Model name	Nodes (*n*)	Conductors (*c*)
I100.DAT	105	I1	64	1846
I200.DAT	72	I2	45	1588
I300.DAT	66	I3	43	876
I400.DAT	148	I4	84	3033
E5CS00.DAT	297	E5C	390	5912
E5HS00.DAT	291	E5H	390	5589
ST3400.DAT	163	ST34	14	1861
		Cold model	640	14895
		Hot model	640	14572

All the nodes are diffusion nodes (except for a dozen of boundary nodes). The conductors are either purely radiative (92% of the total) or purely conductive (8%).

Now some measures are given, as well as the predicted values using the models presented above[27].

The following table lists the **preprocessing** time for the submodels. One can see that the predictions of the formula tend to be quite greater than the measured ones.

[27] The models had been developed 6 months before.

File name	File size (bytes)	Declarative size (bytes)	Preprocessing time (s)	Predicted time (s)
I100.DAT	107391	71579	32.2	67.9
I200.DAT	73268	42475	21.6	46.0
I300.DAT	67588	35788	18.5	42.1
I400.DAT	151736	114052	58.5	96.7
E5CS00.DAT	304340	242697	138.3	194.0
E5HS00.DAT	298211	231755	126.4	188.0

The following table lists the **execution time for a steady state run** for both test cases separately, that is, beginning from the same initial state but one using the cold model and the other one the hot model, and calling the *SOLVIT* routine for the solution and *PRNDTB(' ','L,T',CURRENT)* for the output (that is, they are structurally analogue to the skeleton input deck proposed above). In both cases the number of iterations executed in the *SOLVIT* routine was 102.

Model	Execution time (s)	Predicted time (s)
Cold	169 (out of: 170, 180, 147, 198, 147)	177
Hot	151 (out of: 139, 142, 142, 180, 156)	174

The predicted values agree aceptably with the measured ones and are between the given maximum error bounds, although they tend to exceed the real ones, too. One should also notice that the variance is quite high, especially in the cold model. The time per iteration t_i is 1.66 s for the cold model and 1.48 s for the hot one, whilst the predicted ones are around 1.74 s and 1.72 s, respectively.

A **transient** was also prepared for the cold model. The input deck used is exactly the same as for the steady state but *SLFRWD* in one case and *SLFWBK* in the other one are called instead of *SOLVIT* and the control constants *OUTPUT* and *TIMEND* are set accordingly. The time per iteration t_i was 1.17 s whilst the predicted one is 1.69 s; the difference here is bigger compared with the steady state case.

For the transient case, the ratio real time/computer time was around 0.65 (26 minutes computer time for a transient of 1000 seconds real time)[28].

[28] So it did not run in real time. However the new server is, as mentioned, 2.6 times faster and the ratio grows to 1.7, achieving then real time.

In summary, the predictive models developed have a tendency to overestimate the time for big models (that is, they give a conservative estimate), but are still acceptable for our purposes.

9.5 ESATAN v5.5 vs ESATAN v6.3

In terms of performance, ESATAN v6.3 results to be around 5% faster in preprocessing and 15% faster in executing a simulation. This values are the outcome of a set of test runs carried out on both versions using the sample input decks provided with ESATAN v5.5.

LIST OF REFERENCES[29]

■ EM-ESATAN-056: "ESATAN Engineering Manual", Rel. 2.3. Analysis & Verification Section of ESA/ESTEC and Engineering Research Centre of GEC Alsthom, Aug 92.

■ ESA PSS-03-105 Issue 2: "ESATAN User's Manual," Rel. 2.1. ESA Publications Division, ESTEC, Nov 91.

■ Holman, J P: "Heat Transfer," 4th Ed. McGraw-Hill, 1976.

■ Lienhard, J L: "A Heat Transfer Textbook," 2nd Ed. Prentice-Hall, 1987.

■ Lo Galbo, P; Bouffard B: "SOHO — A Cooperative Scientific Mission to the Sun." ESA Bulletin 71, Aug 92.

■ SH-CAS-NT-00041: "SOHO—SVM: Heat Capacity Model Phase C/D2." CASA (División Espacio). 31/03/93.

■ SH-CAS-NT-00042: "SOHO—SVM: Radiative Model Phase C/D." CASA (División Espacio). 26/03/93.

■ SH-CAS-NT-00043: "SOHO—SVM: Conductive Model Phase C/D2." CASA (División Espacio). 28/06/93.

■ TM-ESATAN-008: "ESATAN Training Manual," Rel. 4.4. Analysis & Verification Section of ESA/ESTEC and Engineering Research Centre of GEC Alsthom, Oct 92.

■ Wood, T D, James, C D: "Design aspects of the attitude control system for the SOHO spacecraft." British Aerospace (Space Systems) Ltd. *Proc. First ESA Int. Conf. on 'Spacecraft Guidance, Navigation and Control Systems'*, ESTEC, 4-7 June 1991 (ESA SP-323, Dec 91).

[29] The mentioned releases for the User's Manual, Engineering Manual and Training Manual are all compatible with ESATAN package version 5.

This page is left blank intentionally for double-sided printing

APPENDIX A: Regression Fits

Given the set of data $(y_i,\boldsymbol{x}_i)$ where $y_i \in \mathbb{R}$, $\boldsymbol{x}_i \in \mathbb{R}^m$, for $i = 1,...,n$, a least-squares regression fit of y as a function of $\boldsymbol{x} = (x_1,...,x_n)$ and with parameters $\boldsymbol{p} = (p_1,...,p_l)$ can be expressed as:

$$y^*(\boldsymbol{x},\boldsymbol{p})$$

$$\min_{\boldsymbol{p}\in\mathbb{R}^l} \overline{E^2}(\boldsymbol{p}) = \min_{\boldsymbol{p}} \left\{ \frac{1}{n}\sum_{i=1}^{n} \left(y_i - y_i^*\right)^2 \right\} = \min_{\boldsymbol{p}} \left\{ \frac{1}{n}\sum_{i=1}^{n} \left(y_i - y^*(\boldsymbol{x}_i,\boldsymbol{p})\right)^2 \right\}$$

Under some conditions one can take partial derivatives of E^2 with respect to each parameter, set them equal to 0 and solve the resulting system of l equations with l unknowns $p_1,...,p_l$[30]:

$$\frac{\partial \overline{E^2}}{\partial p_j} = 0,\ j = 1,\ldots,l$$

Some particular fits of use for this report are given below, along with the system whose solution gives the optimum values for the parameters:

● Linear bidimensional fit: $y^*(x_1,x_2; A,B,C) = Ax_1 + Bx_2 + C$. The optimum values of A, B and C satisfy the system

$$\begin{pmatrix} 20 & 11 & 10 \\ 11 & 02 & 01 \\ 10 & 01 & 00 \end{pmatrix} \begin{pmatrix} A \\ B \\ C \end{pmatrix} = \begin{pmatrix} \overline{x_1 y} \\ \overline{x_2 y} \\ \overline{y} \end{pmatrix}$$

● Linear unidimensional fit: $y^*(x; A,B) = Ax + B$.

[30] Another approach to solve the regression fit is to optimise the objective function

$$f(\boldsymbol{p}) = \frac{1}{n}\sum_{i=1}^{n} \left(y_i - y^*(\boldsymbol{x}_i,\boldsymbol{p})\right)^2$$

in the p-space. A variety of algorithms for unconstrained multiparameter optimisation (non-linear programming) exists, such as the Hooke & Jeeves method, PARTAN etc. Many of them can use a Fibonacci search to minimise the number of function evaluations, which proves to be very valuable in our case, as computing $t(n,c,i)$ means editing the input deck (to set the values for *RELXCA* and *NLOOP*), preprocessing it, running the executable file four or five times (to get a reliable average) and inspecting the corresponding LOG files to find out the elapsed CPU time.

$$\begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} = \begin{pmatrix} \overline{xy} \\ \overline{y} \end{pmatrix}$$

● Quadratic unidimensional fit: $y^*(x; A,B,C) = Ax^2 + Bx + C$.

$$\begin{pmatrix} 4 & 3 & 2 \\ 3 & 2 & 1 \\ 2 & 1 & 0 \end{pmatrix} \begin{pmatrix} A \\ B \\ C \end{pmatrix} = \begin{pmatrix} \overline{x^2y} \\ \overline{xy} \\ \overline{y} \end{pmatrix}$$

● $y^*(x_1,x_2; A,B,C,D) = Ax_2 + Bx_1^2 + Cx_1 + D$.

$$\begin{pmatrix} 02 & 21 & 11 & 01 \\ 21 & 40 & 30 & 20 \\ 11 & 30 & 20 & 10 \\ 01 & 20 & 10 & 00 \end{pmatrix} \begin{pmatrix} A \\ B \\ C \\ D \end{pmatrix} = \begin{pmatrix} \overline{x_2y} \\ \overline{x_1^2y} \\ \overline{x_1y} \\ \overline{y} \end{pmatrix}$$

☞ We use the following compact notation for the arithmetical mean of a cross-product (in a 2D space): for $\alpha,\beta \in \{1,2\}$, $\alpha \neq \beta$, $a,b \in \mathbb{N} \cup \{0\}$,

$$\overline{x_\alpha^a x_\beta^b} \triangleq \frac{1}{n}\sum_{i=1}^{n} x_{\alpha i}^a x_{\beta i}^b \triangleq ab$$

For example

$$\overline{x_1^2 x_2} \triangleq \frac{1}{n}\sum_{i=1}^{n} x_{1i}^2 x_{2i} \triangleq 21$$

The error indicators we use are:

$$\textit{Mean relative error: } \bar{\varepsilon} \triangleq \frac{1}{n}\sum_{i=1}^{n} \left| \frac{y_i - y_i^*}{y_i} \right|$$

$$\textit{Maximum relative error: } \varepsilon_{\max} \triangleq \max_{i=1,\ldots,n} \left\{ \left| \frac{y_i - y_i^*}{y_i} \right| \right\}$$

Be aware that they are just indicators. If we say that the maximum relative error

equals k, we are saying that for all the points measured, the relative error calculated a posteriori will not exceed k. But one has to take into account that it is not the same a relative error of 15% in a running time of 120 hundredths of second (where 18 hundredths of second are hidden by the background noise) as in 30 seconds (where 4.5 seconds make a difference).

Furthermore, those indicators are not the minimisation criterion used in the least-squares method.

APPENDIX B: Experimental Data

Table II.1: (s_i,d_i,t_i) where $t(s,d)$ is the preprocessing time of an input deck with s bytes, d of which belong to definition sentences.

The final value for $t(s,d)$ was calculated as the average of several different runs of the preprocessor on the same input deck.

Model	s	d	t	t^*
FLOWPIPE	2392	1340	6.5	5.79
HEATER	48827	42847	31.0	35.97
MATRIX	11320	1997	9.6	9.81
NLBAR	7293	1050	6.9	7.84
SAVEXX	26572	21950	31.0	21.26
STEADY	2053	295	3.5	5.39
TESTCASE	3773	1013	4.6	6.31
TGMODAL	3898	2593	8.0	6.75
TGTRAN	12057	2652	10.2	10.28
LBAR	1028	475	3.1	4.99

Table II.2: for each model available the running time t is given for different combinations of n, c and i (no. of nodes, conductors and iterations, respectively).

All the values for t, t_i and t_o are in hundredths of second.

The values of *NLBAR* were discarded for the fit of t_i as they are clear outliers and the same happened to the values of c different from 1161 of *HEATER* for the fit of t_o.

Model name	n	c	(i,t)	t_i	t_o
HEATER	168	1161	(10,273) (50,890) (100,1494) (150,2260) (200,2959)	14.0	143
		700	(10,214) (50,630) (100,1108) (150,1646) (200,2148)	10.2	111
		337	(10,181) (50,416) (100,714) (150,1054) (200,1351)	6.2	110
		100	(10,151) (50,278) (100,444) (150,638) (200,814)	3.52	106
SAVEXX	208	615	(3,210) (54,690) (135,1478) (285,2972)	9.81	167
		407	(3,188) (54,538) (135,1169) (285,2211)	7.21	167
		305	(3,161) (54,493) (135,982) (285,1960)	6.36	140
		140	(3,160) (54,376) (135,749) (285,1435)	4.54	139
FLOWPIPE	26	51	(10,105) (50,136) (75,154) (100,168)	0.709	99.3
		43	(10,105) (50,130) (75,145) (100,160)	0.621	98.6
		31	(10,104) (50,123) (75,139) (100,151)	0.531	97.7
		19	(10,102) (50,119) (75,133) (100,144)	0.475	96.4
STEADY	6	8	(10,101) (50,112) (250,122) (1250,231) (6250,821)	0.115	97
LBAR	6	6	(10,91) (100,97) (252,117) (1000,216) (5000,719)	0.126	87.1
NLBAR	6	11	(10,92) (30,89) (50,97) (100,122) (150,128) (200,150) (250,156) (408,196)	0.275	87.5
TESTCASE	11	27	(100,125) (1000,460) (3000,1150) (5000,1890)	0.358	91.2
		20	(100,122) (1000,413) (3000,1027) (5000,1590)	0.3	107
		14	(100,118) (1000,375) (3000,930) (5000,1400)	0.262	109
TGMODAL	58	91	(5,118) (25,159) (100,253) (125,313) (200,448) (307,536)	1.43	124
		65	(5,118) (25,130) (100,223) (125,260) (200,350) (307,466)	1.18	108
		53	(5,116) (25,134) (100,210) (125,240) (200,332) (307,438)	1.08	107
		30	(5,107) (25,135) (100,194) (125,209) (200,294) (307,380)	0.9	106
TGTRAN	58	91	(5,108) (25,132) (100,270) (125,275) (200,363) (307,512)	1.32	109
EVAP2	99	153	(1,115) (10,137) (30,193) (50,232)	2.43	114
MATRIX	8	7	(3,93) (100,109) (500,181) (1000,248) (5000,846)	0.15	97.5

APPENDIX C: Fit Formulae for the Performance Evaluation

Preprocessing time

Being *s* the size in bytes of the input deck and *d* the size in bytes of its declarative part, the preprocessing time *t* in seconds for it can be approximated with a mean relative error of 25% and a maximum relative error of 60% as

$$t^*(s,d) = As + Bd + C$$

$A = 4.3 \cdot 10^{-4}$ s/byte $B = 2.4 \cdot 10^{-4}$ s/byte $C = 4.43$ s

Running time for our skeleton input deck

For an ESATAN skeleton input deck, let *n* be the number of nodes, *c* the number of conductors and *i* the number of iterations required. Then its running time in seconds can be approximated with a mean relative error of 5% and a maximum relative error of 16% as

$$t(n,c,i) = (\alpha c + \beta n^2 + \gamma n + \delta) i + \varepsilon n + \zeta c + \eta$$

where $\alpha = 10^{-4}$, $\beta = 4 \ -10^{-7}$, $\gamma = 7.5 \ -10^{-5}$, $\delta = 0$, $\varepsilon = 2.6 \ -10^{-3}$, $\zeta = 1.2 \ -10^{-4}$, $\eta = 0.93$ (all the parameters are in seconds).

Real-time capability for a simplified transient state simulation scheme

Being t_i the CPU time per iteration, *n* the number of nodes, *c* the number of conductors, $\bar{I}$ the average number of iterations per timestep, *DTIMEU* the length in seconds of the timestep and *MIPS* the speed processor in MIPS, the relation

$$\frac{t_i(n,c) \times \bar{I}}{DTIMEU} \times \frac{8.4}{MIPS} \leq 1$$

should be satisfied with *some* slack for real time to be achieved.