N88-16402

AUTOMATIC MATHEMATICAL MODELING for SPACE APPLICATION

Caroline K. Wang

Software and Data Management Division
Information and Electronic Systems Laboratory
Science and Engineering Directorate
Marshall Space Flight Center/ NASA
Huntsville, Alabama

I. ABSTRACT

This paper describes a methodology for Automatic Mathematical modeling. The major objective is to create a very friendly environment for engineers to design, maintain and verify their model and also automatically convert the Mathematical model into FORTRAN code for conventional computation.

A demonstration program was designed for modeling the Space Shuttle Main Engine Simulation Mathematic Model called Propulsion System Automatic Modeling (PSAM). PSAM is written in LISP and MACSYMA and runs on a Symbolics 3670 Lisp Machine. PSAM provides a very friendly and well organized environment for engineers to build a knowledge base for base equations and general information. PSAM contains an initial set of component process elements for the Space Shuttle Main Engine Simulation and a questionnaire that allows the engineer to answer a set of questions to specify a particular model. PSAM is then able to automatically generate the model and FORTRAN code. A future goal is to download the FORTRAN code to the VAX/VMS system for conventional computation.

II. INTRODUCTION

Mathematical Modeling for Space Application Simulationproject is a very complicated process which includes Analysis, Design, and the Generation of complex equations. Generally the model will require several modification before it will match a real system. Historically the modifications have been time consuming and a fertile source of error.

The use of Artificial Intelligence techniques has shown that this process can be simplified. This paper describes a methodology for organizing and generating the model automatically.

III. EXPERT USER INTERFACE and AUTOMATIC KNOWLEDGE BASE GENERATION

An Artificial Intelligence system can provide a very friendly work environment for engineers. Through PSAM's questionnaire, the system collects the necessary information and generates the knowledge base automatically.

The knowledge base includes: (1) An information base in frames and
(2) Generic equations for the project.

IV. SOFTWARE TOOLS and AUTOMATIC EQUATION AND CODE GENERATION

There are a number of tools available for equation derivation and FORTRAN code generation. The one we choose to use is "MACSYMA". LISP was the language I used basically for user interface and knowledge base generation. I also used LISP for generating "MACSYMA" code. The MACSYMA code generates the equations automatically and at the same time FORTRANcode will also be generated automatically and saved into the disk file for future integration.

The future goal is to integrate the FORTRAN code we generated and download to the VAX/VMS system for conventional computation.

V. MAINTAINING AND VERIFYING

One of the most difficult problems in software today is the verification and maintenance of existing programs, especially programs built up our time with many programmers involved. This is particularly true for simulation programs. The traditional way of modifying simulation programs by rebuilding the model and recoding the model had a potential source of many errors.

The automatic system can eliminates most of these problems. The proce-dure become much simpler which makes it easy for the user to maintain and modify the model and program directly.

VI. EXAMPLE

A demonstration program was designed for modeling the Space Shuttle Main Engine Simulation Mathematic Model called Propulsion System Automatic Modeling (PSAM). PSAM is written in LISP and MACSYMA and runs on a Symbolics 3670 LISP machine.

The design goals for PSAM were to develop automatic modeling skills for Propulsion System, and other scientific and Engineering applica-tions. We used the old Engine Model for an example to study.

PSAM includes the following features:

(1). Friendly user interface.
(2). Automatic Knowledge Base generation and
(3). Automatic Equation and Coding generation.

The Space Shuttle Main Engine Simulation model was built up from the component process elements and their combination into the sub-programs.

The component process elements are Pump,Hot gas turbine, Hydraulic turbine, Turbopump, Combustor, Valve, Incompressible propellant flow, Injector volume with priming for start, Hot gas heat transfer and Regen cooling flow. The subprograms are Fuel, Oxidizer and Hot Gas.

There are two types of information for a PSAM knowledge base. One is the component process elements generic equations and the other is the information base for the combination of the Space Shuttle Main Engine model subprograms and component process elements.

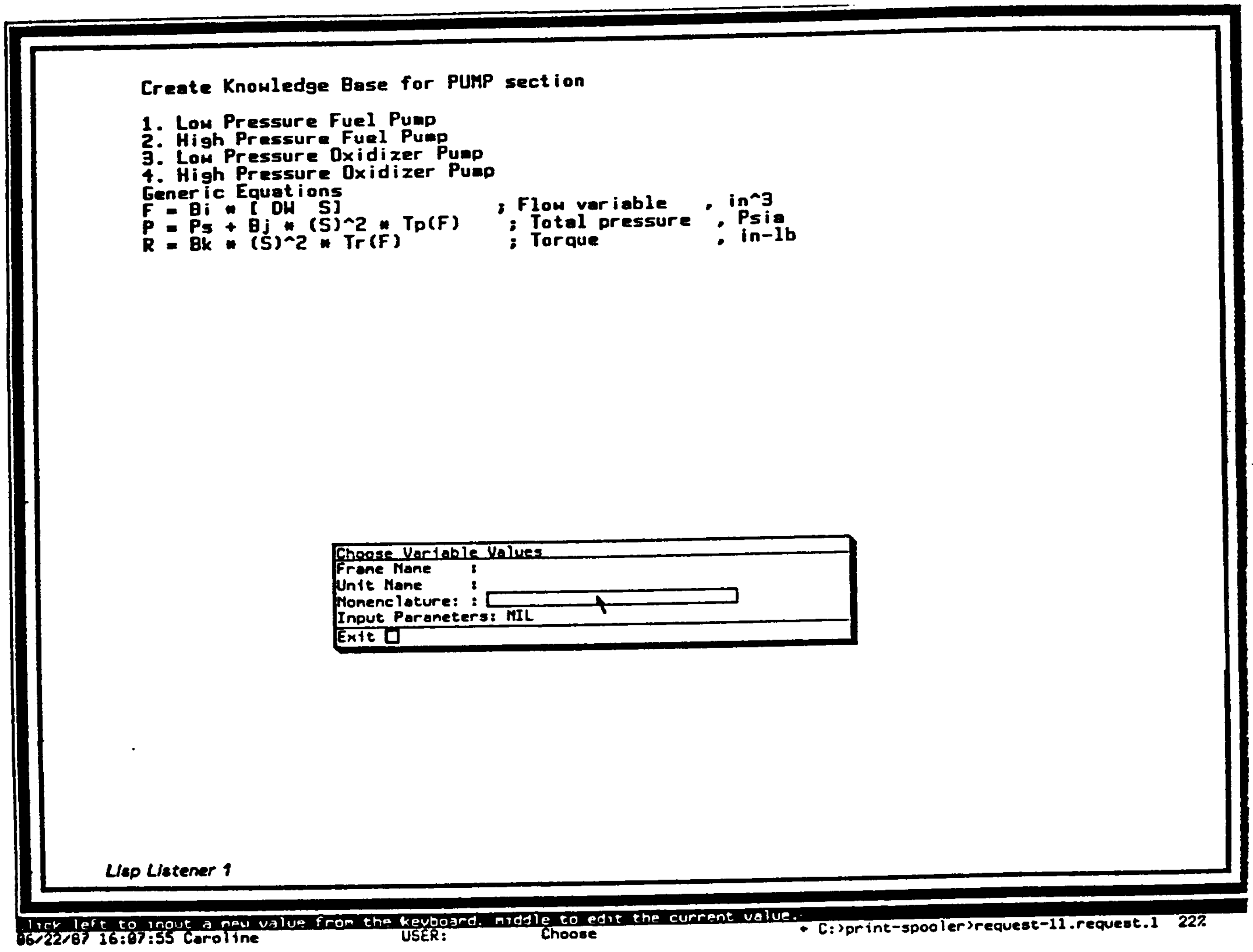
The Expert System collects the detailed requirements and generates

the sets of specific equations for the component process elements and subprograms.

PSAM has the ability to:

(1) Create or maintain the Knowledge base
(2) Load different knowledge base
(3) Automatically generate Equations
(4) Output generated Equation or FORTRAN code to Disk file or
    option for print out of the Laser printer.

Example for input information user interface for the component process element Pump knowledge base.



```
Create Knowledge Base for PUMP section

1. Low Pressure Fuel Pump
2. High Pressure Fuel Pump
3. Low Pressure Oxidizer Pump
4. High Pressure Oxidizer Pump
Generic Equations
F = Bi * [ DW  S]                ; Flow variable    , in^3
P = Ps + Bj * (S)^2 * Tp(F)       ; Total pressure   , Psia
R = Bk * (S)^2 * Tr(F)            ; Torque           , in-lb
```

```
Choose Variable Values
Frame Name    :
Unit Name     :
Nomenclature: :
Input Parameters: NIL
Exit
```

```
Lisp Listener 1
lick left to input a new value from the keyboard, middle to edit the current value.
06/22/87 16:07:55 Caroline        USER:        Choose              • C:>print-spooler>request-11.request.1   22%
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
(Deframe Ffpl
          (Class PUMP)
          (Unit "flow-variable")
          (Nomenclature "Low Pressure Fuel Pump Flow variable")
          (Input-parameter Bll DWfdl Sfl))
(Deframe Ffp2
          (Class PUMP)
          (Unit "flow-variable")
          (Nomenclature "High Pressure Fuel Pump Flow Variable")
          (Input-parameter B18 DWfd2 Sf2))
(Deframe Fdpl
          (Class PUMP)
          (Unit "flow-variable")
          (Nomenclature "Low Pressure Oxidizer Pump Flow variable")
          (Input-parameter B27 DWmov+DWop3 Sol))
(Deframe Fop2
          (Class PUMP)
          (Unit "flow-variable")
          (Nomenclature "High Pressure Oxidizer Pump Flow variable")
          (Input-parameter B39 DWmov+DWotl+DWop3 So2))
(Deframe Pfdl
          (Class PUMP)
          (Unit "total-pressure")
          (Nomenclature "Low Pressure Fuel Pump Discharge Total pressure")
          (Input-parameter Pfs B12 Sfl Tpfpl))
(Deframe Pfd2
          (Class PUMP)
          (Unit "total-pressure")
          (Nomenclature "High Pressure Fuel Pump Total Pressure")
          (Input-parameter Pfdl B19 Sf2 Tpfp2))
(Deframe Podl
          (Class PUMP)
          (Unit "total-pressure")
          (Nomenclature "Low Pressure Oxidizer Pump Total Pressure")
          (Input-parameter Pos B28 SØl Tpopl))
(Deframe Pod2
          (Class PUMP)
          (Unit "total-pressure")
          (Nomenclature "High Pressure Oxidizer Pump Total Pressure")
          (Input-parameter Podl B4Ø So2 Tpop2))
(Deframe Rfpl
          (Class PUMP)
          (Unit "torque")
          (Nomenclature "Low Pressure Fuel Pump "torque"")
          (Input-parameter B13 Sfl Trfpl))
(Deframe Rfp2
          (Class PUMP)
          (Unit "torque")
          (Nomenclature "High Pressure Fuel Pump "torque"")
          (Input-parameter B2Ø Sf2 Trfp2))
(Deframe Ropl
          (Class PUMP)
          (Unit "torque")
          (Nomenclature "Low Pressure Oxidizer Pump "torque"")
          (Input-parameter B34 Sol Tropl))
```

PSAM Generic equations:

```
"**********PUMP*************"$


"Pump Flow Variable"$

PFV(F,B,DW,S):=
Block ( F = B * (DW /S));

"Pump Total Pressure"$

PTP(PP,P,B,S,T):=
Block ( PP = P + B * (S)^2 * T);

"Pump Torque"$

PT(R,B,S,T):=
Block ( R = B * (S)^2 * T);


"**********TURBINE**********"$


"Turbine Torque"$

TT(R,B,P,T):=
Block ( R = B * P * T);

"Turbine Speed Parameter"$

TSP(M,B,S,T):=
Block ( M = B * S / (T^(1/2)));

"Turbine Weight Flowrate"$

TWF(DW,B,dP,r):=
Block ( DW = (B * (dP) * r)^(1/2));

"**********TURBOPUMP SPEED*******"$

"TurboPump Speed"$

TPS(S,B,Tq,SØ):=
Block ( S = B * 'Integrate ( Tq,t) + SØ);


"**********VALVE**********"$


"Valve Area"$

VA(A,Ab,T,Th,Thb):=
Block ( A = Ab * (T * Th / Thb));
```

```
"Valve Total Pressure Inlet Total"$

VTPIT(P,Pi,RF,DW,Rho):=
Block ( P = Pi - RF * (DW)^2 /Rho);

"Valve Total Pressure Inlet Static"$

VTPIS(PP,P,DW,A,Rho):=
Block ( PP = P - DW^2 /(772.8 * A^2 * Rho));

"Valve Delta Total Pressure"$

VDTP(DP,B,Rhob,Rho,DW,A,Ab,RF):=
Block ( DP = B * (Rhob/Rho)*(DW/(A/Ab))^2 - RF * DW /Rho);


"**********FLOW**********"$


"Fuel Flow"$

FF(DWf,Bi,P9,P,Bj,R,DWØ):=
Block ( DWf = B * 'integrate (((P9 - P - (Bi / R)) * DWf^2),t) + DWØ);

"Oxidizer Flow"$

OF(DWo,Bk,Ppos,P,Bl,DWØ,A,Ab,Bm,DWi):=
Block( DWo = Bk * 'integrate ((Ppos - p - Bl * (DWo /(A/Ab))^2 -
Bm*(DWi)^2),t) + DWØ);

"Variable in injector Priming function, Dimensionless"$

VIPF(E,Bh,DWo,DWi,EØ):=
Block( E = Bh * 'integrate ((DWo - DWi),t) + EØ);

"Weight Flowrate Injector"$

WFI(DWi,DWo,Eo):=
Block( DWi = DWo * Eo);


"**********COMBUSTER**********"$


"Combuster Total Pressure"$

CTP(P,Bi,DW1,DW2,Bj,DW3,PØ):=Block( P = Bi * 'integrate (( DW1 + DW2 - Bj *
DW3),t) + pØ);

"Combuster Fuel Weight Flowrate"$
```

```
CFWF(F,DW2,DW):=Block( F = DW2 / (DW1 + DW2));

"Combuster Temperature"$

CT(T,R,F,Bk,T9):=Block ( T = R(F) + Bk * T9);


"**********COOLING ELEMENT**********"$


"Cooling Element Total Pressure"$

CETP(P,B,DQw1,DQw2,H3,DWi,H,DW,PØ):=
Block( P = B * 'integrate((DQw1 + DQw2 + H3 * DWi - H * DW),t) + PØ);

"Cooling Element Specific Enthalpy"$

CESE(H,B,T):=Block( H = B * T);

"Cooling Element Weight Flowrate Main Chamber Heat Exchanger"$

CEWFM(DW,Bi,P,P5,Bj,R5,DW,DWØ):=
Block( DW = Bi * 'integrate ((P - P5 - (Bj / R5) * DW^2),t) +DWØ);

"Cooling Element Density"$

CED(R,B,DWi,DW,RØ):=
Block( R = B * 'integrate ((DWi - DW),t) + RØ);

"Cooling Element Temperature"$

CET(T,B,P,R):=
Block( T = B * ( P / R));

"Cooling Element Heat Transfer Rate Hot Gas Wall"$

CEHTRHGW(DQw1,B,T,Tw1,DWi):=
Block( DQw1 = B * (1.Ø + Ø.ØØ2 * T) * (DWi)^Ø.8);

"Cooling Element Heat Transfer Rate Ambient Thrust Chamber"$

CEHTRAT(DQw2,B,Tw1,T,DWi):=
Block( DQw2 = B * (1.Ø + Ø.ØØ2 * T) * (Tw2 - T) * DWi^Ø.8);

"Cooling Element Heat Transfer Rate TC"$

CEHTR(DQtc,B,Tc,Tw1,DWcn):=
Block( DQtc = B * (Tc - Tw1) * DWcn^Ø.8);

"Cooling Element Hot Gas Wall Temperature"$

CEHGWT(T,DQtc,DQw1,TØ):=
Block( T = B * 'integrate ((DQtc - DQw1) * temp(t),t) + TØ);
```

The Macsyma code that psam generates which will call the Generic equation function blocks and also generates the equations and FORTRAN code for the certain element.

```
fpump():=block(writefile("al:>psam>equation>pump-equation."),
display(PT(ROP1,B34,SO1,TROP1)),
display(PT(RFP2,B2Ø,SF2,TRFP2)),
display(PT(RFP1,B13,SF1,TRFP1)),
display(PTP(POD2,POD1,B4Ø,SO2,TPOP2)),
display(PTP(POD1,POS,B28,SØ1,TPOP1)),
display(PTP(PFD2,PFD1,B19,SF2,TPFP2)),
display(PTP(PFD1,PFS,B12,SF1,TPFP1)),
display(PFV(FOP2,B39,DWMOV+DWOT1+DWOP3,SO2)),
display(PFV(FDP1,B27,DWMOV+DWOP3,SO1)),
display(PFV(FFP2,B18,DWFD2,SF2)),
display(PFV(FFP1,B11,DWFD1,SF1)),
closefile(),
writefile("al:>psam>fortran>pump.for"),
fortran(PT(ROP1,B34,SO1,TROP1)),
fortran(PT(RFP2,B2Ø,SF2,TRFP2)),
fortran(PT(RFP1,B13,SF1,TRFP1)),
fortran(PTP(POD2,POD1,B4Ø,SO2,TPOP2)),
fortran(PTP(POD1,POS,B28,SØ1,TPOP1)),
fortran(PTP(PFD2,PFD1,B19,SF2,TPFP2)),
fortran(PTP(PFD1,PFS,B12,SF1,TPFP1)),
fortran(PFV(FOP2,B39,DWMOV+DWOT1+DWOP3,SO2)),
fortran(PFV(FDP1,B27,DWMOV+DWOP3,SO1)),
fortran(PFV(FFP2,B18,DWFD2,SF2)),
fortran(PFV(FFP1,B11,DWFD1,SF1)),
closefile());
```

$$ROP1 = B34\ SO1^2\ TROP1$$

$$RFP2 = B2Ø\ SF2^2\ TRFP2$$

$$RFP1 = B13\ SF1^2\ TRFP1$$

$$POD2 = B4Ø\ SO2^2\ TPOP2 + POD1$$

$$POD1 = B28\ SØ1^2\ TPOP1 + POS$$

$$PFD2 = B19\ SF2^2\ TPFP2 + PFD1$$

$$PFD1 = B12\ SF1^2\ TPFP1 + PFS$$

$$FOP2 = \frac{B39\ (DWOT1 + DWOP3 + DWMOV)}{SO2}$$

$$FDP1 = \frac{B27\ (DWOP3 + DWMOV)}{SO1}$$

$$FFP2 = \frac{B18\ DWFD2}{SF2}$$

$$FFP1 = \frac{B11\ DWFD1}{SF1}$$

```
ROP1  = B34*SO1**2*TROP1
RFP2  = B2Ø*SF2**2*TRFP2
RFP1  = B13*SF1**2*TRFP1
POD2  = B4Ø*SO2**2*TPOP2+POD1
POD1  = B28*SØ1**2*TPOP1+POS
PFD2  = B19*SF2**2*TPFP2+PFD1
PFD1  = B12*SF1**2*TPFP1+PFS
FOP2  = B39*(DWOT1+DWOP3+DWMOV)/SO2
FDP1  = B27*(DWOP3+DWMOV)/SO1
FFP2  = B18*DWFD2/SF2
FFP1  = B11*DWFD1/SF1
```

The sets or the FORTRAN code will then be integrated into a completed compiled SSME program and downloaded to VAX/VMS system for conventional computation.