MODELLING AND OPTIMISATION OF DISTRIBUTED PARAMETER SYSTEMS WITH ECOSIMPRO

Carmen G Moles, Antonio A Alonso and Julio R Banga*

Process Engineering Group, IIM-CSIC
c/ Eduardo Cabello 6, 36208, Vigo, Spain

Abstract

This work describes in detail the steps followed to be able to model distributed parameter systems with EcosimPro. To achieve this, macros were created to implement the NMOL for one and two spatial dimensions. We have also exploited the sparse matrix of the resulting set of DAEs after spatial discretisation, which reduces the necessary computation times for the simulation of systems described by partial differential equations.

Key Words: EcosimPro, dynamic simulation, partial differential equations, distributed systems.

1 INTRODUCTION

EcosimPro® is a simulation tool programmed with a high-level language (EL) which enables declarations of systems, and even discrete events, to be made simply and intuitively by means of DAEs. It does not, however, enable direct declarations of systems formulated by partial differential equations (PDEs or, in general, PDAEs).

This work therefore addresses the implementation of the numerical method of lines (NMOL) for one and two dimensions in the EcosimPro environment (more details on the NMOL can be found in books written by Schiesser [6], [7]).

To determine an initial approximation we developed the NMOL by defining the functions in EL. In spite of the correct behaviour observed in the sparse systems which we implemented, the necessary computation time quickly increased with the system dimension.

To overcome this difficulty, we incorporated macros in place of the functions and this considerably reduced the computation time of each simulation. This is basically due to the fact that, through the macros, EcosimPro has access to the complete symbolic structure of the DAEs resulting from NMOL discretisation, allowing it to be adequately exploited by means of integrators which use sparse systems.

The following sections briefly address the basics of the NMOL and show its implementation and use in EcosimPro. Finally, two examples are given to show the advantages of using macros.

2 DESCRIPTION OF THE NUMERICAL METHOD OF LINES

The numerical method of lines, or NMOL [6], is a technique for solving equations in parabolic or hyperbolic partial differential equations (PDEs) by transforming them into a set of ODEs or DAEs which can be solved with a standard integrator. It is based on the discretisation of the space domain using different methodologies which employ uniform, nonuniform or adaptive meshing [3], [4].

The value of the dependent variable at one point of the mesh is calculated based on the values at adjacent points, where the number of points used determines the type or order of the method.

To illustrate the process of obtaining these formulae, we will describe the steps followed to obtain the order 2 formula (more details can be found in books by Schiesser [7], [8]) with uniform meshing in space.

Let us assume a single dimensional problem (the results can be easily extrapolated to 2D or 3D cases) $x=x(\xi)$. We will consider that a good approximation of the solution to the PDE at a point can be defined with the Taylor Series:

$$x(\xi) = a_0 + a_1(\xi - \xi_i) + a_2(\xi - \xi_i)^2 + a_3(\xi - \xi_i)^3 + \ldots \quad (1)$$

where:

$$a_n = \frac{1}{n!}\frac{\partial^n x(\xi_i)}{\partial \xi^n} \quad (2)$$

If we use a mesh of equidistant σ elements and a formula to calculate the value of the field at two nonconsecutive points of the mesh we obtain:

$$x(\xi_{i-1}) = x(\xi_i) + \frac{\partial x(\xi_i)}{\partial \xi}(\xi_{i-1} - \xi_i) + \frac{1}{2!}\frac{\partial^2 x(\xi_i)}{\partial \xi^2}(\xi_{i-1} - \xi_i)^2 + ..$$
$$x(\xi_{i+1}) = x(\xi_i) + \frac{\partial x(\xi_i)}{\partial \xi}(\xi_{i+1} - \xi_i) + \frac{1}{2!}\frac{\partial^2 x(\xi_i)}{\partial \xi^2}(\xi_{i+1} - \xi_i)^2 + .. \quad (3)$$

and subtracting the above two expressions we get:

$$\frac{\partial x(\xi_i)}{\partial \xi} = \frac{x(\xi_{i+1}) - x(\xi_{i-1})}{2\Delta\xi} + O(\Delta\xi^2) \quad (4)$$

This enables us to directly calculate the value of the space derivative as a function of the field values at two nonconsecutive points of the mesh. However, this expression is only valid for i=2,....., σ-1. For the first and last points of the mesh we have to calculate other expressions; therefore, to approximate the value of the space derivative at point ξ_1 we use the field values at points ξ_2 and ξ_3:

$$x(\xi_2) = x(\xi_1) + \frac{\partial x(\xi_1)}{\partial \xi}\Delta\xi + \frac{1}{2!}\frac{\partial^2 x(\xi_2)}{\partial \xi^2}(\Delta\xi)^2 + ...$$
$$x(\xi_3) = x(\xi_2) + \frac{\partial x(\xi_1)}{\partial \xi}2\Delta\xi + \frac{1}{2!}\frac{\partial^2 x(\xi_2)}{\partial \xi^2}(2\Delta\xi)^2 + \quad (5)$$

subtracting:

$$\frac{\partial x(\xi_1)}{\partial \xi} = \frac{-3x(\xi_1) + 4x(\xi_2) - x(\xi_3)}{2\Delta\xi} + O(\Delta\xi^2) \quad (6)$$

Likewise, calculating the approximations for $\xi_{\pi-2}$ and $\xi_{\pi-1}$ and subtracting, we obtain the following formula:

$$\frac{\partial x(\xi_\pi)}{\partial \xi} = \frac{3x(\xi_\pi) - 4x(\xi_{\pi-1}) + x(\xi_{\pi-2})}{2\Delta\xi} + O(\Delta\xi^2) \quad (7)$$

We now have an approximation of the first space derivative for each point of the mesh.
A similar procedure is followed to obtain formulae for other orders.

Second order formulae (of 3 points) expressed compactly would be:

$$\frac{d\tilde{x}}{d\xi} = \frac{1}{2\Delta\xi}\begin{Bmatrix} -3 & 4 & -1 \\ -1 & 0 & 1 \\ 1 & -4 & 3 \end{Bmatrix}\tilde{x} \quad (8)$$

The second order space derivatives can be calculated by the recursive application of these formulae. By direct substitution in the original PDE we obtain a set of ODEs, one for each node of the mesh. The resulting number of ODEs therefore depends on the space discretisation level selected, which noticeably increases as the nonlinearity and space dimension of the problem increases.

In this work we have chosen not to recursively use first order formulae to calculate second order space derivatives. The numerical method of lines has been implemented using second order formulae (3 points) to calculate both the first and second order space derivatives, the formulae for which can be easily obtained following a procedure similar to that described above.

3 IMPLEMENTATION OF MACROS IN ECOSIMPRO

A macro is an instruction in a programming language that is replaced by a sequence of instructions which are coded only once, assigning them a new name, and which can be used as many times as necessary (and recursively). During execution, each macro is substituted by the coded sequence of instructions, acting on possible parameters defined as arguments.

The EcosimPro macros were created by Pedro Cobas (EA Internacional) in response to our need to implement the NMOL. The use of macros involves the pre-processing of the file containing the component (called, for example, fichero.el). This pre-processing expands the macros, defined in a file called macros.h, creating a new EL file (eg, newfile.el) which has to be compiled in a second step in the usual way. After the pre-processing, the structure and particularly the sparse properties of the equations (DAEs) generated by the NMOL are ready to be adequately exploited by sparse integrators.

To validate the macros in the EcosimPro environment, different previously validated models were used with real experimental data. The values of the model parameters were established and the output values were suitably stored for subsequent representation and analysis.

4 DEFINITION OF MACROS

The macros are declared in a C++ header (called, for example, macros.h). Note: to be able to use these macros one of the EcosimPro libraries has to be substituted. Contact Pedro Cobas for more information.

- This file is saved to a directory which can be accessed by EcosimPro

The only requirement for its use is its inclusion at the beginning of the component using:

#include<macro dir>/macros.h

As an example, let us introduce the code which implements the NMOL in one dimension when no boundary conditions are specified:

```
// Create the term dfx for the different finites:

#define dfx(xL,xU,N) \
  (1.0 / (2.0*((xU-xL) / (N -1))))

// Calculate the first derivative of ξ in ξd without imposing extreme conditions:

#define PDE_1D(xL,xU, N, ξ, ξd) \

ξd[1]=dfx(xL,xU,N)*(-3*ξ[1]+4.0*ξ[2]-1.0
*ξ[3]) &\

EXPAND (i IN 2,N-1) &\
ξd[i]=dfx(xL,xU,N)* (-ξ[i-1] + ξ[i+1]) &\

ξd[N]=dfx(xL,xU,N)*(ξ[N-2]-4.0*ξ[N-1]+ 3.0*ξ[N])
```

where xL is the lower limit of x (dimension), xU is the upper limit of x, N is the number of intervals of the domain (including the boundary points), ξ is the variable to be differentiated and ξd is the first derivative.

5 FORMULATION OF THE NMOL FOR ONE DIMENSION IN ECOSIMPRO

By implementing the NMOL in EcosimPro we have considered the existence of different types of boundary conditions such as Dirichlet and Neumann, and also that such boundary conditions are not specified. In each case, the so-called NMOL from the corresponding EcosimPro component undergoes small variations such as:

- *Calculation of the first derivative of ξ in x (dimension) without imposing extreme conditions:*

PDE_1D(xL, xU, N, ξ, ξd)

where xL is the lower limit of x, xU is the upper limit of x, N is the number of intervals of the domain (including the boundary points), ξ is the variable to be differentiated and ξd is the first derivative.

- *Calculation of the first and second derivative ξ without imposing extreme conditions:*

PDE_1D_2der(xL, xU, N, ξ, ξd, ξdd)

where ξdd is the second derivative of the variable ξ.

- *Calculation of the first derivative of ξ imposing extreme conditions:*

PDE_1D_EXTR(xL, xU, N, ξ, ξd, flag1, ux1, flagN, uxN)

where flag1 indicates whether second order boundary conditions (Neumann) exist on border 1 (TRUE if affirmative, FALSE if negative) and flagN indicates whether second order boundary conditions exist on border N.

In addition, ux1 is the value of the boundary condition on border 1 and uxN on border N.

- *Calculation of the first and second derivative of ξ imposing extreme conditions:*

PDE_1D_EXTR_2der(xL, xU, N, ξ, ξd, ξdd, flag1, ux1, flagN, uxN)

The different cases are defined in a single macro, thus facilitating its use.

6 FORMULATION OF THE NMOL FOR TWO DIMENSIONS IN ECOSIMPRO

As in the case of one dimension, different calls have been implemented for the NMOL depending on the existing boundary conditions. The main difference in the case of two dimensions is that the dimension with respect to which it is derived must also be specified. The different possibilities are:

- *Calculation of the first derivative of* ξ in x (dimension) *without imposing extreme conditions:*

PDE_2D(xL, xU, N1, N2, flag, ξ, ξd)

where xL is the lower limit of x, xU is the upper limit of x, N1 is the number of intervals of domain 1 (including the boundary points), N2 is the number of intervals of domain 2, flag indicates the dimension with respect to which it is derived (TRUE indicates the first dimension and FALSE the second dimension), ξ is the variable to be differentiated and ξd is the first derivative.

- *Calculation of the first and second derivative of ξ in x without imposing extreme conditions:*

PDE_2D_2der(xL, xU, N1, N2, flag, ξ, ξd, ξdd)

where ξdd is the second derivative calculated numerically.

- Calculation of the first derivative of ξ imposing extreme conditions:

PDE_2D_EXTR(xL, xU, N1, N2,flag, ξ, ξd, flag1, ux1, flagN, uxN)

where flag1 indicates whether second order boundary conditions exist on border 1 and flagN indicates whether second order boundary conditions exist on border N1 or N2, depending on the dimension with respect to which it is derived.

- Calculation of the first and second derivative of ξ in x imposing extreme conditions:

PDE_2D_EXTR_2der(xL, xU, N1, N2, flag, ξ, ξd, ξdd, flag1, ux1,flagN, uxN)

In each case, and depending on the characteristics of the problem (whether it has boundary conditions and of what order), the call will be made to one function or the other.

7 EXAMPLES

A large number of the processes that take place in the food and biotechnological industry are characterised by the fact that they are formulated by means of partial differential equations.
Two examples have been selected –namely, the sterilisation of foods in cylindrical cans and the aseptic processing of fluid media- to show how any sparse system would be implemented by means of macros in EcosimPro. We also point out the advantages of implementing the NMOL by means of macros with respect to the use of functions.

7.1 STERILISATION OF FOOD IN A CYLINDRICAL CAN (TWO-DIMENSIONAL PROBLEM)

We consider the sterilisation of very viscous solid or liquid foods (exclusively conduction-heated) which are canned. We used the Banga and Co mathematical model [2] which consists of:

- Heat transmission equation in non-steady state (Fourier)

$$\frac{dT}{\partial t} = \alpha\left(\frac{\partial^2 T}{\partial r^2} + \frac{1}{r}\frac{\partial T}{\partial r} + \frac{\partial^2 T}{\partial z^2}\right)$$
$$\frac{dF_c}{dt} = 10^{(T(0,0)-T^*_{microorg})/Z_{microorg}} \qquad (9)$$

where T is the temperature of the food inside the can.

- Kinetics of thermal destruction of micro-organism spores and kinetics of thermal degradation of nutrients and/or organoleptic factors

$$\frac{dX(r,z)}{dt} = -\frac{2.3025}{D^*_{microorg}} X(r,z)\cdot e^{2.3025(T(r,z)-T^*_{microorg})/Z_{microorg}}$$
$$\frac{dS(r,z)}{dt} = -\frac{2.3025}{D^*_{nutrient}} S(r,z)\cdot e^{2.3025(T(r,z)-T^*_{nutrient})/Z_{nutrient}}$$

where X and S is the concentration of micro-organisms and nutrients, respectively.

- Evaluation of the lethality and final mean nutrient and micro-organism concentration

$$X_{media} = \frac{2}{V_t}\int_0^R\int_0^L rX(r,z)dr\cdot dz$$
$$S_{media} = \frac{2}{V_t}\int_0^R\int_0^L rS(r,z)dr\cdot dz \qquad (11)$$

$$F_S = -D^*_{microorg}\log\left(\frac{X_{media}}{X_0}\right) \qquad (12)$$

where F_c is the lethality at the critical point and F_s is the total lethality, integrated into the whole can. Finally, X_{media} y S_{media} are the mean values of micro-organism and nutrient concentration.

In addition, to correctly define the model, the following boundary conditions are fixed:

$$\frac{\partial T(r,L)}{\partial z} = T_{retort}$$
$$\frac{\partial T(R,z)}{\partial r} = T_{retort}$$
$$\frac{\partial T(0,z)}{\partial r} = 0$$
$$\frac{\partial T(r,0)}{\partial z} = 0 \qquad (13)$$

In order to illustrate the implementation of the component, we include an extract of the EL code corresponding to the sterilisation model, focusing on implementing the non-steady state heat transmission equation and the calculation of the PDEs:

```
#include "\macros.h"
COMPONENT EstCilind

CONTINUOUS

-- First and second derivative with respect to r:
-- Boundary conditions:
```

```
EXPAND(i IN 1,NR) N1UxN[i] = 0
EXPAND(i IN 1,NR) N1Ux1[i] = 0

-- Call to NMOL (macro):
PDE_2D_EXTR_2der(0,R,NR,NL,TRUE,
T,Tr,Trr,TRUE,N1Ux1,FALSE,N1Ux)

-- First and second derivative with respect to r:
-- Boundary conditions:

EXPAND(i IN 1,NL) N2UxN[i] = 0
EXPAND(i IN 1,NL) N2Ux1[i] = 0

-- Call to NMOL (macro):
PDE_2D_EXTR_2der(0,L,NR,NL,FALSE,
T,Tz,Tzz,TRUE,N2Ux1,FALSE,N2UxN)

-- Fourier's Law:
EXPAND(j IN 1,NL-1)
T[1,j]'= alfa*(2.0*Trr[1,j] + Tzz[1,j])

EXPAND(i IN 2,NR-1)
 EXPAND(j IN 1,NL-1)
    T[i,j]'=alfa*(Trr[i,j]+((1/((i-1)*DR))*
                Tr[i,j]) +Tzz[i,j])
```

After expanding the differential equations with the NMOL, the system is formed by:

Table 1: Description of the Model

Equations	Total V	Explicit V	Dynamic V
1462	1462	1119	343

Figures 1 and 2 show that simulation of the heat process using the original functions (in blue) and the new MACROS (in red) lead to exactly the same results.

Figure 1: Representation of lethality at the critical point F_c and total lethality F_s (integrated into the whole can)

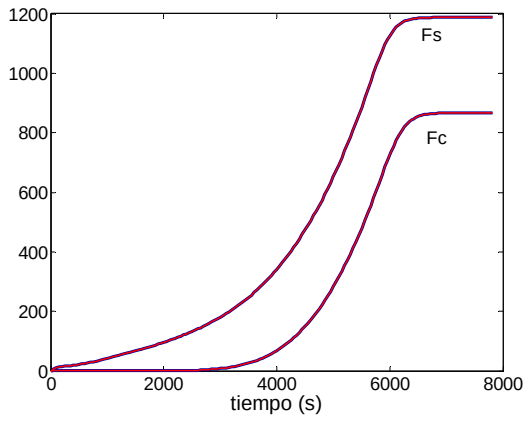
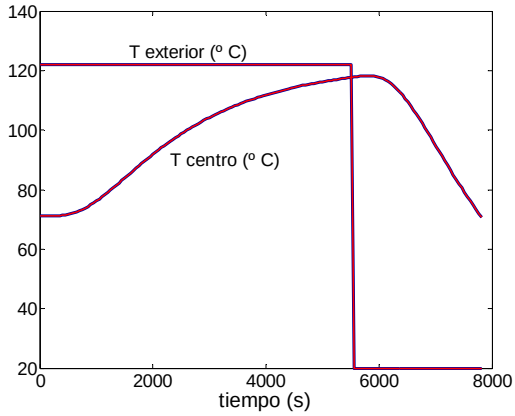


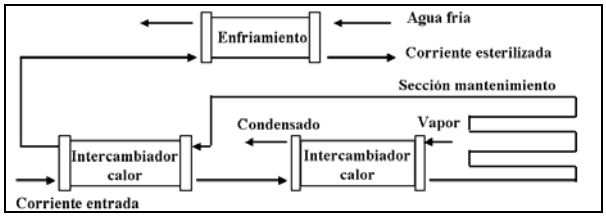
Figure 2: Evolution of the temperature on the outside and in the centre of the cylindrical can



7.2 ASEPTIC PROCESSING

The second example shows the simulation of aseptic processing. The continuous steriliser comprises two heat exchangers and a temperature holding section (see Figure 3) based on the model proposed by Armenante and Leskowicz [1]. In the first heat exchanger, the food flow is preheated by indirect contact with the sterile flow from the temperature holding section (energy saving). In the second heat exchanger, the preheated flow reaches sterilisation temperature by indirect contact with steam. Finally, the sterilisation temperature is maintained in the temperature holding section for the length of time necessary to obtain the desired reduction of micro-organisms.

Figure 3: Sketch of Aseptic Process



The temperature of the fluid medium is a function of time and the temperature of round particles is a function of position.

Bacillus stearothermophilus has been taken as the reference contaminating micro-organism based on its resistance to heat sterilisation (fixing a safe sterilisation level). This reduction must be achieved in the fluid flow by means of the contribution of the temperature holding section alone (although in reality there is overall contribution from the other heat exchangers). The model was implemented in EcosimPro using the NMOL implemented through

macros. We finally obtain a system formed by 2955 equations and 672 dynamic variables (for more details, see table 2).

Table 2: Description of the Model

Equations	Total V	Explicit V	Dynamic V
2955	2955	2283	672

Owing to the extension of the model, we have only included the energy and material balances of the fluid and solid phase in the heat holding section, where:

HEAT HOLDING SECTION (Fluid Medium)
Material balance:

$$\frac{4v_f}{\pi D^2}\frac{\partial \log N(z)}{\partial z} = -\exp(\log K_{d0} - \frac{E_d}{RT(z)})$$
$$\frac{4v_f}{\pi D^2}\frac{\partial \log C(z)}{\partial z} = -\exp(\log K_{d0\_nut} - \frac{E_{d\_nut}}{RT(z)}) \quad (14)$$

where v_f is the volumetric flow, $\log K_{d0}$ and $\log K_{d0\_nut}$ are the kinetic constants of destruction (1st order) for the cells and nutrients, E_{d_nut} and E_d are the micro-organism and nutrient activation energies, R is the universal constant of the gases, logN and logC are the micro-organism and nutrient concentrations, T is the temperature of the fluid and D is the diameter of conduction.

Energy balance:

$$(1-x_i)v_f \cdot \rho \cdot C_p \frac{\partial T(z)}{\partial z} =$$
$$Q(z) - \frac{3}{4}x_i \pi D^2 / Rd \cdot h(T(z) - T_s(z,1)) \quad (15)$$

where ÷ is the density of the fluid, C_p is its calorific capacity, h is the heat transfer coefficient, Rd is the radius of the particles, T and T_s are the temperatures of the fluid and the particles, and Q is the heat exchanged.

HEAT HOLDING SECTION (Solid particles)
Material balance:

$$\frac{4v_f}{\pi D^2}\frac{\partial \log N_s(z,r)}{\partial z} = -\exp(\log K_{d0} - \frac{E_d}{RT_s(z,r)}) \quad (16)$$

where $\log N_s$ is the concentration of micro-organisms in the particles.

Energy balance:

$$\frac{4v_f}{\pi D^2}\frac{\partial T_s(z,r)}{\partial z} = \frac{k_s}{\rho_s C_{ps} Rd^2}(\frac{\partial^2 T_s(z,r)}{\partial r^2} + \frac{2}{r}\frac{\partial T_s(z,r)}{\partial r})$$
$$\frac{\partial T_s(z,0)}{\partial r} = 0 \quad (17)$$
$$k_s \frac{\partial T_s(z,1)}{\partial r} = Rd \cdot h(T(z) - T_s(z,1))$$

where T_s is the temperature of the particles, C_{ps} is the calorific capacity of the particles, ÷$_s$ is their density, k_s is their heat conductivity and D is the diameter of conduction.

Implementation of this model (described by partial differential equations) in EcosimPro is partly shown in an extract of the material and energy balances of the solid particles in the heat holding section.

```
#include "\macros.h"
COMPONENT ProcAseptico

CONTINUOUS

--HOLDING SECTION (Solid particles)
-- First derivative with respect to z:

PDE_2D(0,Length,NLength,NRadius,TRUE,logNs,
logNsz)

-- First derivative with respect to z:

PDE_2D(0,Length,NLength,NRadius,TRUE,Ts,Tsz)

-- First derivative with respect to r:
-- Boundary conditions:
   EXPAND(i IN 1,NLength)
   N2TsN[i] = ((h*Radius)/ks)*(T[i]-Ts[i,NRadius])

   EXPAND(i IN 1,NLength)
   N2Ts1[i] = 0.0

-- Second derivative with respect to r:
 PDE_2D_EXTR_2der(0,RadiusValue,NLength,
 NRadius,FALSE,Ts,Tsr,Tsrr,TRUE,N2Ts1,TRUE,
 N2TsN)

-- Mass balance:

EXPAND(i IN 2,NLength)
 EXPAND(j IN 1,NRadius)

logNs[i,j]'+(vf*4/pi/Diameter**2)*
logNsz[i,j]= -exp(logKdOEd/(R*Ts[i,j]))

-- Energy balance:
 EXPAND(i IN 2,NLength)
   Ts[i,1]'+(vf*4/pi/Diameter**2)*Tsz[i,1]=
      (ks/(rhos*cps*Radius**2))*(3*Tsrr[i,1])
```

```
EXPAND(i IN 2,NLength)
 EXPAND(j IN 2,NRadius)

 Ts[i,j]'+(vf*4/pi/Diameter**2)*Tsz[i,j]=
ks/(rhos*cps*Radius**2)*(Tsrr[i,j]+
(2/((j-1)*DR))*Tsr[i,j])
```

8 IMPROVEMENTS OBTAINED WITH THE MACROS

The main and most important improvement is the considerable reduction in CPU time required to simulate the models when the DASSL_SPARSE integrator is selected and sparse information generated internally by EcosimPro is used.

However, no reduction in CPU time is achieved with the other integrators (DASSL, RK4) owing to that explained previously.

Table 3 shows the results (on a PIII/450 MHz) for food sterilisation in a two-dimensional cylindrical can (example taken from Banga and Co [2]). It is worth noting the significant reduction in simulation time (from 55.06 to 7.84 seconds) when we use the macros and the DASSL_SPARSE integrator.

Table 3: Comparison of CPU times (in seconds) implementing the NMOL as EcosimPro functions and as macros

	DASSL SPARSE	**DASSL**	**RK4**
NMOL FUNCTION	55.06	70.07	0.42
MACRO	7.84	69.58	0.43

9 CONCLUSIONS

In this paper we have presented the implementation through macros of the numerical method of lines (NMOL). This procedure facilitates easy declaration in EcosimPro of systems described by partial differential equations.

This tool can be hugely effective and useful when we want to implement and simulate real processes described by different partial differential equations which, after discretisation, can involve thousands of DAEs. In such cases, the reduction in the CPU time required to execute each simulation will be a critical factor and we have demonstrated how to deal with it with the use of sparse integrators.

Acknowledgements

We wish to thank Pedro Cobas for all his help in the development of this work.

References

[1] Armenante, P., Leskowicz, M., (1990) Design of continuous sterilization systems for fermentation media containing suspended solids, Biotechnology Progress, 6, pp. 292-306.

[2] Banga, J.R., Martín, R.P., Gallardo, J.M. and Casares, J.J., (1991) "Optimization of thermal processing of conduction-heated canned foods: Study of several objective functions", J. Food Engineering, 14(1), pp. 25-51.

[3] Li, S., (1998) Adaptive Mesh Methods and Software for Time-Dependent Partial Differential Equations, PhD thesis, Univ. of Minnesota, USA.

[4] Li, S. and Petzold, L. (1999) Design of new {DASPK} for sensitivity analysis, Technical report, UCSB.

[5] Oh, M. (1995) Modelling and Simulation of Combined Lumped and Distributed Processes, PhD thesis, Department of Chemical Engineering and Chemical Technology. Imperial College of Science, Technology and Medicine.

[6] Schiesser, W.E., (1991) The Numerical Method of Lines, Academic Press, New York.

[7] Schiesser, W.E., (1994) Computational Mathematics in Engineering and Applied Science: ODEs, DAEs and PDEs, CRC Press, Inc.

[8] Silebi, C. and Schiesser, W. (1992) Dynamic Modeling of Transport Process Systems, Academic Press, London.