Using Simulink S-Functions with Finite Difference Method Applied for Heat Exchangers

STEPAN OZANA, MARTIN PIES
Department of Measurement and Control
VSB-Technical University of Ostrava
17.listopadu 15/2172
CZECH REPUBLIC
stepan.ozana@.vsb.cz, martin.pies@.vsb.cz http://kat455.vsb.cz

Abstract: - This paper deals with application of finite difference method for solving a general set of partial differential equations in Matlab&Simulink environment. Particularly it describes use of Simulink S-functions which make it possible to set-up the most complex systems with complicated dynamics. There's a comparison of M and C S-functions which are two main approaches when building user-defined blocks in Simulink, regarding the performance and efficiency of the simulation for M and C versions of the codes and possibility to perform a real-time simulation. As an example, these approaches are shown on solving the dynamics of a concurrent and a counter-flow heat exchanger.

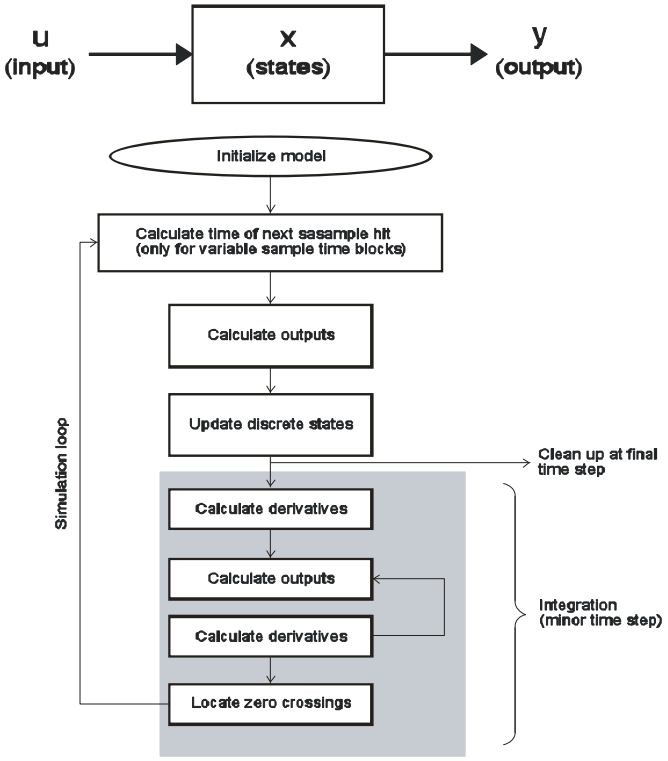
Key-Words: - S-function, Matlab, Simulink, heat exchanger, partial differential equations, finite difference method

1 Introduction

S-functions (system-functions) provide a powerful mechanism for extending the capabilities of Simulink. This introductory paragraph describes what S-function is and when and why it is convenient to use one. S-functions make it possible to add customized algorithms to Simulink models, either written in MATLAB or C. By following a set of simple rules it possible to implement the algorithms in an S-function. After S-function has been written and placed its name in an S-Function block (available in the *User-defined Functions* sublibrary), it's time to customize the user interface by using masking. An S-function is a computer language description of a dynamic system. S-functions can be written using MATLAB or C. C language S-functions are compiled as MEX-files using the mex utility described in the Application Program Interface Guide. As with otherMEX-files, they are dynamically linked into MATLAB when needed. S-functions use a special calling syntax that enables you to interact with Simulink's equation solvers. This interaction is very similar to the interaction that takes place between the solvers and built-in Simulink blocks. The form of an S-function is very general and can accommodate continuous, discrete, and hybrid systems. As a result, nearly all Simulink models can be described as S-functions. The most common use of S-functions is to create custom Simulink blocks. S-functions can be effectively used for a variety of applications, including:

•Adding new general purpose blocks to Simulink
•Incorporating existing C code into a simulation
•Describing a system as a mathematical set of equations
•Using graphical animations

An advantage of using S-functions is that it is possible to build a general purpose block that can be used many times in a model, varying parameters with each instance of the block.



An M-file or a CMEX-file that defines an S-Function block must provide information about the model; Simulink needs this information during simulation. As the simulation proceeds, Simulink, the ODE solver, and the M-file interact to perform specific tasks. These tasks include defining initial conditions and block

characteristics, and computing derivatives, discrete states, and outputs. Simulink provides a template M-file S-function that includes statements that define necessary functions, as well as comments to help with writing the code needed for a particular S-function block.

M-file S-functions work by making a sequence of calls to S-function routines, which are M-code functions that perform tasks required by customized S-function. This table lists the S-function routines available to M-file S-functions. C MEX-file S-functions have the same structure and perform the same functions as M-file S-functions. In addition, C MEX S-functions provides more functionality than M-file S-functions.

2 Flow Scheme of Heat Exchangers

The paper tackles the problem of simplified model of heat exchangers (concurrent and counter-flow), described by three state variables as stated on Fig.1. and Fig.2.

$T_1(x,t)$ temperature of steam

$T_2(x,t)$ temperature of flue gas

$T_S(x,t)$ temperature of the wall

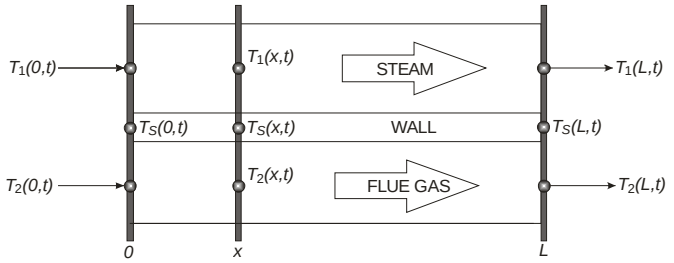


Fig.1 Physical state variables of a concurrent heat exchanger

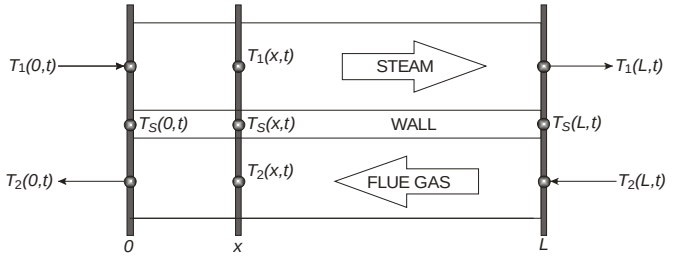


Fig.2 Physical state variables of a counter-flow heat exchanger

For a concurrent heat exchanger, the main goal of this task is to determine temperatures of a steam and a flue gas $T_1(L,t)$ and $T_2(L,t)$ as reactions on inlet temperatures $T_1(0,t)$ and $T_2(0,t)$. As for a counter-flow heat exchanger, it is determination of temperatures of a steam and a flue gas $T_1(L,t)$ and $T_2(0,t)$ as reactions on temperatures $T_1(0,t)$ and $T_2(L,t)$.

3 Simplified Mathematical Model of Heat Exchanger

Due to the fact that heat exchangers are systems with distributed parameters, mathematical model of concurrent and counter-flow heat exchangers is described by a set of partial differential equations (1-3):

$$T_S - T_1 = \tau_1\left[u_1\frac{\partial T_1}{\partial x} + \frac{\partial T_1}{\partial t}\right] \quad (1)$$

$$T_S - T_2 = \tau_2\left[u_2\frac{\partial T_2}{\partial x} + \frac{\partial T_2}{\partial t}\right] \quad (2)$$

$$\frac{T_1 - T_S}{\tau_{S1}} + \frac{T_2 - T_S}{\tau_{S2}} = \frac{\partial T_S}{\partial t} \quad (3)$$

where T_1, u_1 stands for temperature and velocity of a steam, T_2, u_2 is temperature and velocity of a flue gas and T_S stands for wall temperature. τ_1, τ_2, τ_s are time constants of the system. For purposes of applying finite difference method by means of Simulink S-functions, particular partial derivatives of state variables must be expressed as follows:

$$\frac{\partial T_1}{\partial t} = \frac{1}{\tau_1}(T_S - T_1) - u_1\frac{\partial T_1}{\partial x} \quad (4)$$

$$\frac{\partial T_2}{\partial t} = \frac{1}{\tau_2}(T_S - T_2) - u_2\frac{\partial T_2}{\partial x} \quad (5)$$

$$\frac{\partial T_S}{\partial t} = \frac{1}{\tau_{S1}}(T_1 - T_S) + \frac{1}{\tau_{S2}}(T_2 - T_S) \quad (6)$$

4 Finite Difference Method

Finite difference method uses approximation of partial derivatives by numerical differences in particular points using the following formulas, omitting the remainders:

$$\left.\frac{\partial T_1(x,t)}{\partial x}\right|_{x=x_1} \approx \frac{-3T_1(x_1,t) + 4T_1(x_2,t) - T_1(x_3,t)}{2h} \quad (7)$$

$$\left.\frac{\partial T_1(x,t)}{\partial x}\right|_{x=x_2} \approx \frac{T_1(x_3,t) - T_1(x_1,t)}{2h} \quad (8)$$

$$\left.\frac{\partial T_1(x,t)}{\partial x}\right|_{x=x_3} \approx \frac{T_1(x_4,t) - T_1(x_2,t)}{2h} \quad (9)$$

$$\left.\frac{\partial T_1(x,t)}{\partial x}\right|_{x=x_n} \approx \frac{T_1(x_{n-2},t) - 4T_1(x_{n-1},t) + 3T_1(x_n,t)}{2h} \quad (10)$$

Graphical representation of approximation formulas can be seen on Fig.3, showing approximation of $T_1(x,t)$.The same concept is then used for temperatures $T_2(x,t), T_S(x,t)$.

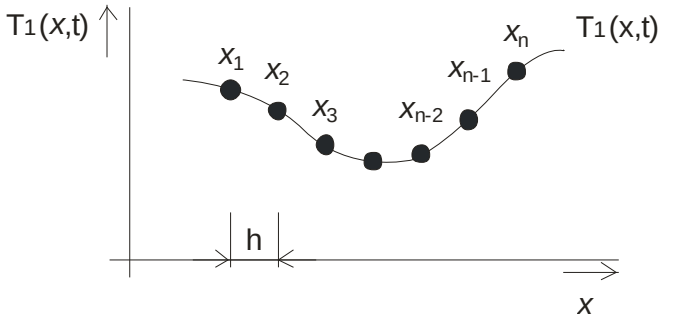


Fig.3 Example of approximation of partial derivatives by finite differences

5 Concurrent Heat Exchanger

This paragraph describes use of finite difference method applied on a concurrent heat exchanger in detail, assuming C version of S-function. First, the length L is divided into N slices along x-axis according the scheme on Fig.4.

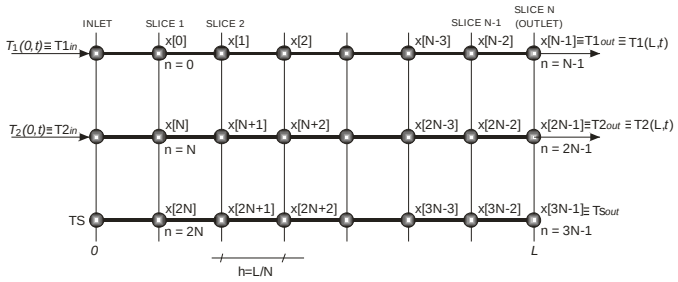


Fig.4 Notation of finite differences for state variables

Notice: All of the following equations are stated for C S-functions, the first index starts from zero. For M version, the syntax is a bit different, round brackets are used for indexes and the first index starts with one. In accordance with Fig.4, the following formulas hold for state variables, their derivatives and outputs:

$$T_1: \quad x[0] \;\div\; x[N-1]$$
$$T_2: \quad x[N] \;\div\; x[2N-1]$$
$$T_S: \quad x[2N] \div\; x[3N-1]$$
$$\frac{dT_1}{dt}: \quad dx[0] \;\div\; dx[N-1]$$
$$\frac{dT_2}{dt}: \quad dx[N] \;\div\; dx[2N-1]$$
$$\frac{dT_S}{dt}: \quad dx[2N] \div\; dx[3N-1]$$

$$T_{1out} \equiv x[N-1]$$
$$T_{2out} \equiv x[2N-1]$$

Applying formulas (7-9) on a set of equations (4-6), the following relations are obtained for the state derivatives:

SLICE 1 ($n = 0$):

$$dx[0] = \frac{1}{\tau_1}\left(x[2N] - x[0]\right) - u_1 \frac{x[1] - T_{1in}}{2h}$$

$$dx[N] = \frac{1}{\tau_2}\left(x[2N] - x[N]\right) - u_2 \frac{x[N+1] - T_{2in}}{2h}$$

$$dx[2N] = \frac{1}{\tau_{S1}}\left(x[0] - x[2N]\right) + \frac{1}{\tau_{S2}}\left(x[N] - x[2N]\right)$$

Due to the formulas (7-9), the first and the last slice must be treated separately. For inner slices, there is a loop containing the formulas as follows:

SLICE 2 to N-1 ($n = 1 \div (N-2)$):

$$dx[n] = \frac{1}{\tau_1}\left(x[n+2N] - x[n]\right) - u_1 \frac{x[n+1] - x[n-1]}{2h}$$

$$dx[n+N] = \frac{1}{\tau_2}\left(x[n+2N] - x[n+N]\right) - u_2 \frac{x[n+N+1] - x[n+2N]}{2h}$$

$$dx[n+2N] = \frac{1}{\tau_{S1}}\left(x[n] - x[n+2N]\right) + \frac{1}{\tau_{S2}}\left(x[n+N] - x[n+2N]\right)$$

SLICE N (outlet, $n = N-1$):

$$dx[N-1] = \frac{1}{\tau_1}\left(x[3N-1] - x[N-1]\right) - u_1 \frac{x[N-3] - 4x[N-2] + 3x[N-1]}{2h}$$

$$dx[2N-1] = \frac{1}{\tau_2}\left(x[3N-1] - x[2N-1]\right) - u_2 \frac{x[2N-3] - 4x[2N-2] + 3x[2N-1]}{2h}$$

$$dx[3N-1]=\frac{1}{\tau_{S1}}\left(x[N-1]-x[3N-1]\right)+$$

$$+\frac{1}{\tau_{S2}}\left(x[2N-1]-x[3N-1]\right)$$

The following code shows crucial parts of C code of particular S-function in Simulink:

Method *outputs*:

```
y[0]=x[N-1];       //outlet, temperature T1 (steam)
y[1]=x[2*N-1];   //outlet, temperature T2 (flue gas)
```

Method *derivatives*:

```
h=l/N;
//slice 1  , n=0
dx[0] = 1/tau1*(x[2*N]-x[0])-u1/(2*h)*(x[1]-
*T1_vst[0]);
dx[N] = 1/tau2*(x[2*N] - x[N]) - u2/(2*h)*(x[N+1] -
*T2_vst[0]);
dx[2*N] = 1/taus1*(x[0] - x[2*N]) + 1/taus2*(x[N] -
x[2*N]);
//slice 2 to N-1, n=1 to N-2
for (n = 1; n <= N-2; n++) {
dx[n]=1/tau1*(x[n+2*N]-x[n])-u1/(2*h)*(x[n+1]-x[n-
1]);
dx[n+N]=1/tau2*(x[n+2*N]-x[n+N])-
u2/(2*h)*(x[n+N+1]-x[n+N-1]);
dx[n+2*N]=1/taus1*(x[n]-
x[n+2*N])+1/taus2*(x[n+N]-x[n+2*N]); }
//slice N, n=N-1
dx[N-1]=1/tau1*(x[3*N-1]-x[N-1])-u1/(2*h)*(x[N-3]-
4*x[N-2]+3*x[N-1]);
dx[2*N-1]=1/tau2*(x[3*N-1]-x[2*N-1])-
u2/(2*h)*(x[2*N-3]-4*x[2*N-2]+3*x[2*N-1]);
dx[3*N-1]=1/taus1*(x[N-1]-x[3*N-
1])+1/taus2*(x[2*N-1]-x[3*N-1]);
```

6 Counter-flow Heat Exchanger

This paragraph describes use of finite difference method applied on a counter-flow heat exchanger in detail, assuming C version of S-function. First, the length L is divided into N slices along x-axis according the scheme on Fig.5, compare with Fig.4.

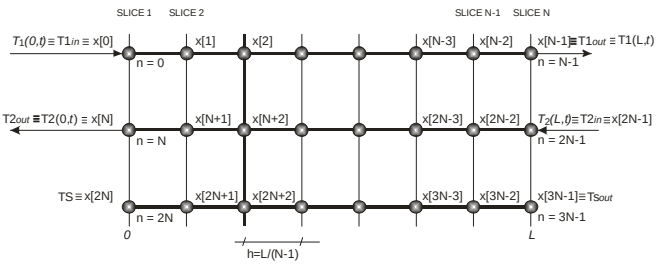


Fig.5 Notation of finite differences for state variables applied for a counter-flow heat exchanger

The following formulas hold for state variables, their derivatives and outputs:

$$T_1:\quad x[0]\ \div\ x[N-1]$$

$$T_2:\quad x[N]\ \div\ x[2N-1]$$

$$T_S:\quad x[2N]\div\ x[3N-1]$$

$$\frac{dT_1}{dt}:\quad dx[0]\ \div\ dx[N-1]$$

$$\frac{dT_2}{dt}:\quad dx[N]\ \div\ dx[2N-1]$$

$$\frac{dT_S}{dt}:\quad dx[2N]\div\ dx[3N-1]$$

$$T_{1out}\equiv x[N-1]$$

$$T_{2out}\equiv x[N]$$

Applying formulas (7-9) on a set of equations (4-6), the following relations are obtained for the state derivatives:

SLICE 1 (n = 0):

$$dx[0]=\frac{1}{\tau_1}\left(x[2N]-T_{1in}\right)$$

$$-u_1\frac{-3T_{1in}x+4x[1]-x[2]}{2h}$$

$$dx[N]=\frac{1}{\tau_2}\left(x[2N]-x[N]\right)-$$

$$-u_2\frac{-3x[N]+4x[N+1]-x[N+2]}{2h}$$

$$dx[2N]=\frac{1}{\tau_{S1}}\left(T_{1in}-x[2N]\right)+$$

$$+\frac{1}{\tau_{S2}}\left(x[N]-x[2N]\right)$$

SLICE 2 (n = 1):

$$dx[1]=\frac{1}{\tau_1}\left(x[2N+1]-x[1]\right)$$

$$-u_1\frac{x[2]-x[0]}{2h}$$

$$dx[N+1]=\frac{1}{\tau_2}\left(x[2N+1]-x[N+1]\right)-$$

$$-u_2\frac{x[N+2]-x[N]}{2h}$$

$$dx[2N+1]=\frac{1}{\tau_{S1}}\left(x[1]-x[2N+1]\right)+$$

$$+\frac{1}{\tau_{S2}}\left(x[N+1]-x[2N+1]\right)$$

SLICE n (n = 2÷(N – 3)):

$$dx[n]=\frac{1}{\tau_1}(x[n+2N]-x[n])-$$
$$-u_1\frac{x[n+1]-x[n-1]}{2h}$$
$$dx[n+N]=\frac{1}{\tau_2}(x[n+2N]-x[n+N])-$$
$$-u_2\frac{x[n+N+1]-x[n+2N]}{2h}$$
$$dx[n+2N]=\frac{1}{\tau_{S1}}(x[n]-x[n+2N])+$$
$$+\frac{1}{\tau_{S2}}(x[n+N]-x[n+2N])$$

SLICE N-1: (n = N – 2)

$$dx[N-2]=\frac{1}{\tau_1}(x[3N-2]-x[N-2])-$$
$$-u_1\frac{x[N-1]-x[N-3]}{2h}$$
$$dx[2N-2]=\frac{1}{\tau_2}(x[3N-2]-x[2N-2])-$$
$$-u_2\frac{x[2N-2]-x[2N-3]}{2h}$$
$$dx[3N-2]=\frac{1}{\tau_{S1}}(x[N-2]-x[3N-2])+$$
$$+\frac{1}{\tau_{S2}}(x[2N-2]-x[3N-2])$$

SLICE N: (right edge, n = N – 1)

$$dx[N-1]=\frac{1}{\tau_1}(x[3N-1]-x[N-1])-$$
$$-u_1\frac{x[N-3]-4x[N-2]+3x[N-1]}{2h}$$
$$dx[2N-1]=\frac{1}{\tau_2}(x[3N-1]-T_{2in})-$$
$$-u_2\frac{x[2N-3]-4x[2N-2]+3T_{2in}}{2h}$$
$$dx[3N-1]=\frac{1}{\tau_{S1}}(x[N-1]-x[3N-1])+$$
$$+\frac{1}{\tau_{S2}}(T_{2in}-x[3N-1])$$

These lines of code show crucial parts of C code of S-function in Simulink:

Method *outputs*:

```
y[0]=x[N-1];      //outlet, temperature T1 (steam)
y[1]=x[N];        //outlet, temperature T2 (flue gas)
```

Method *derivatives*:

```
//first slice
dx[0] = 1/tau1*(x[2*N]- *T1_in[0])-u1*(-3**T1_in[0]
+ 4*x[1] - x[2])/(2*h);
dx[N] = 1/tau2*(x[2*N] - x[N]) - u2*(-3*x[N] +
4*x[N+1] - x[N+2])/(2*h);
dx[2*N] = 1/taus1*(*T1_in[0] - x[2*N]) +
1/taus2*(x[N] - x[2*N]);
//second slice
dx[1] = 1/tau1*(x[2*N+1]- x[1])-u1*(x[2] -
x[0])/(2*h);
dx[N+1] = 1/tau2*(x[2*N+1] - x[N+1]) - u2*(x[N+2] -
x[N])/(2*h);
dx[2*N+1] = 1/taus1*(x[1] - x[2*N+1]) +
1/taus2*(x[N+1] - x[2*N+1]);
//inner slices
   for (n = 2; n <= N-3; n++) {
      dx[n]     = 1/tau1*(x[n+2*N] - x[n])-u1*(x[n+1]-
x[n-1])/(2*h);
      dx[n+N]   = 1/tau2*(x[n+2*N] - x[n+N])-
u2*(x[n+N+1] - x[n+N-1])/(2*h);
      dx[n+2*N] = 1/taus1*(x[n] -
x[n+2*N])+1/taus2*(x[n+N] - x[n+2*N]);
   }
//the slice previous to the last one
dx[N-2]=1/tau1*(x[3*N-2] - x[N-2])-u1*(x[N-1] - x[N-
3])/(2*h);
dx[2*N-2]=1/tau2*(x[3*N-2] - x[2*N-2])-u2*(x[2*N-1]
- x[2*N-3])/(2*h);
dx[3*N-2]=1/taus1*(x[N-2]-x[3*N-
2])+1/taus2*(x[2*N-2] - x[3*N-2]);
//last slice
dx[N-1]=1/tau1*(x[3*N-1] - x[N-1])-u1*(x[N-3] -
4*x[N-2] + 3*x[N-1])/(2*h);
dx[2*N-1]=1/tau2*(x[3*N-1] - *T2_in[0])-u2*(x[2*N-
3] - 4*x[2*N-2] + 3**T2_in[0])/(2*h);
dx[3*N-1]=1/taus1*(x[N-1] - x[3*N-
1])+1/taus2*(*T2_in[0] - x[3*N-1]);
```

7 Comparison of C and M Versions

From the point of view of Simulink, due to its concept and masking block's parameters, there's no difference between working with C and M version of an S-function because the blocks behaves the same, as it is demonstrated on Fig.6.

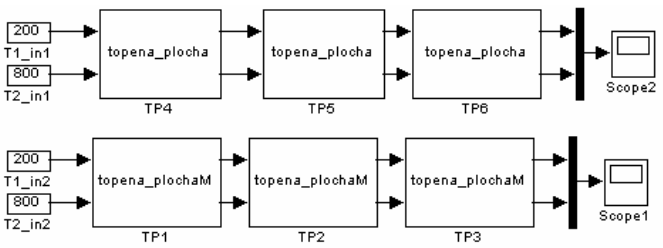


Fig.6 Simulink scheme of M and C versions of S-functions, three heat exchangers set up in series

The syntax of the codes inside the blocks is different. Generally, M structure of S-function has syntax of Matlab language and the abilities are limited. C structure requires a bit of C programming but the performance and effectiveness are uncomparably higher. The comparison was done for simulation of 10 seconds for M and C version, depending on number of blocks in series and number of slices. Particular simulations are also compared to real time.

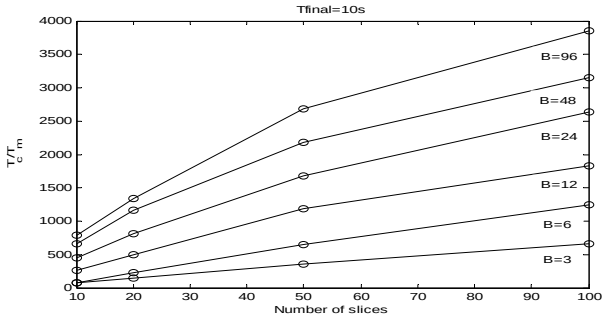


Fig.6 Ratio of performances of C and M S-functions, dependence on number of slices

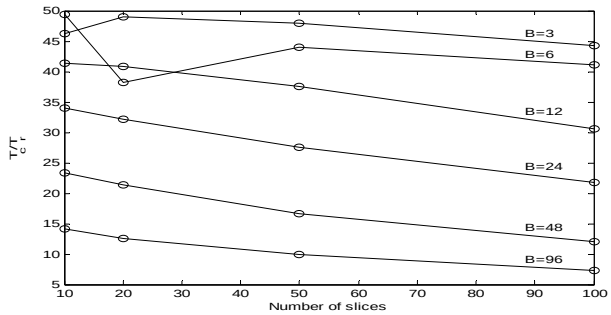


Fig.7 Performance of C S-functions vs. real time

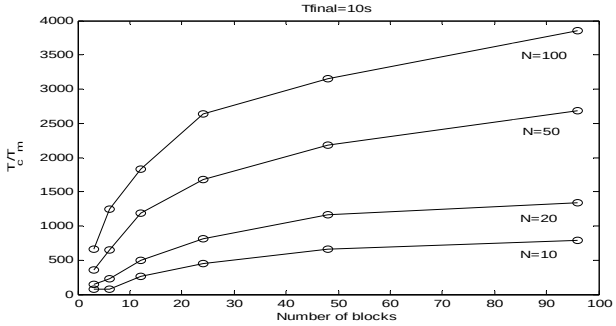


Fig.8 Ratio of performances of C and M S-functions, dependence on number of blocks in series

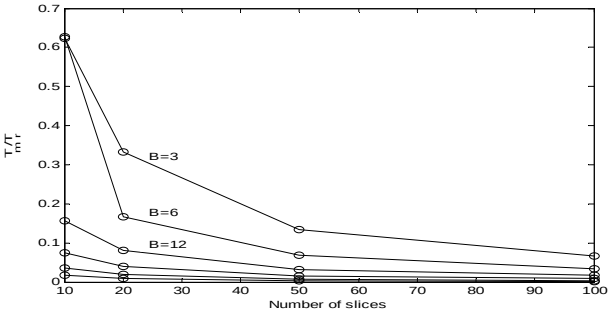


Fig.9 Performance of M S-functions vs. real time

8 Conclusion

The main purpose of this paper was to verify the possibility of use Simulink S-functions to solve the dynamics of simplified model of heat exchangers. The results of these simulations were verified with the data from a real process. Fig.6 and Fig.8 show the ratios between the real time needed for simulation of C code (T_C) and the time needed to perform M code (T_m) for different number of slices (marked as N parameter) and different number of blocks in series (marked as B parameter). The C code is from 100 to almost 4000 times faster then M code. Fig.7 then describes the time needed for simulation of C code vs. real time (T_r). It states that even for a large number of blocks in series and a large number of slices (B=96, N=100) the simulation is still 6 times faster then real time. On the other hand, Fig.9 shows that M S-functions are inconvenient to use for systems with so complicated dynamics and it's impossible to perform a real time simulation because even for a small number of slices and blocks in series (N=3, B=10) the simulation is much slower than real time. Currently, the approach described in this paper has been successfully used and tested for extended (full) mathematical model of the heat exchangers, which is extended by further state and output variables, particularly velocities and pressures of temperatures of steam and flues gas.

Acknowledgement:
The work was supported by the grant "Simulation of heat exchangers with the high temperature working media and application of models for optimal control of heat exchangers", No.102/09/1003, of the Czech Science Foundation.

References:

[1] Cook R. D. at al., Applications of Finite Element Analysis, *John Wiley and Sons, 2002, ISBN 0-471-350605-0*

[2] Haberman R., Applied Partial Differential Equations with Fourier Series and Boundary Value Problems, 4th Edition, *Pearson Books, 2003, ISBN13: 9780130652430 ISBN10: 0130652431*

[3] Hanuš B., Regulační charakteristiky příhřívačů páry u kotlů československé výroby. *Strojírenství 11, 1961, č.3., str. 179-184*

[4] Kattan. P.I., MATLAB Guide to Finite Elements: An Interactive Approach. Seconnd Edition. *Springer New York 2007. ISBN-13 978-3-540-70697-7*

[5] IHI, Steam Generators, *Technical Note of Ishikawajima-Harima Heavy Industries Co., Ltd. Japan, 2004. 381210-11*1000-0403R(Y)*

[6] http://www.mathworks.com

[7] Writing S-functions v.3, Mathworks documentation