

A Neural Network Based Macromodel for Microflow Sensors¹

O. Mikulchenko*, A. Rasmussen** and K. Mayaram***

* School of EECS, Washington State University, ETRL
Room 309, Pullman 99164, WA, USA, mikul@eecs.wsu.edu

**The Institute of MEMS and VLSI Technologies, The George Washington University,
Washington D.C., USA, arasmuss@seas.gwu.edu

***Dept. of ECE, Oregon State University, Corvallis, OR, USA, karti@ece.orst.edu

ABSTRACT

A neural network based macromodel has been developed for a microflow sensor and implemented in the circuit simulator SPICE3f5. The model has been validated with numerical simulations and allows accurate and efficient simulation of microflow sensors in a microfluidic system. This model simulates both the steady-state and the dynamic operation of the flow sensor. The neural network design is based on a problem-based scaling and a combination of stochastic search and the genetic algorithm. The macromodel can be used for microfluidic system simulation and the optimal design of the flow sensor.

Keywords: microflow sensors, macromodeling, neural networks, microfluidic simulation.

1 INTRODUCTION

The modeling of microsystems at a system level requires appropriate compact or macro models for microdevices. In general, it is difficult or almost impossible to obtain closed form analytical models for microdevices. Thus, models are obtained by a simplification of the full physical model. The compact models allow for fast system level simulation of the microsystem. However, the accuracy of such a model can be questionable because of the simplifications made during the model development phase. Model accuracy and simplicity are important for the design of complex systems.

Microflow sensor modeling has been examined by several authors [1-4]. These approaches have used a solution of the partial differential equations (PDEs) for the coupled fluid/thermal problem. The work of [1, 2] employs equivalent circuit descriptions that are solved in SPICE whereas [3, 4] solve the PDEs using the finite-element and the finite-difference methods, respectively. However, none of these methods provide a simple and accurate macromodel for the flow sensor.

In this paper, we present a macromodeling approach for microflow sensors. The macromodel is based on an accurate multivariate approximation of the solutions of the

PDEs by neural networks (NNs). The paper is organized in the following manner. In Section 2, the modeling of the flow sensor is described. The model validations are presented in Section 3. Details of the complete model and implementation are given in Section 4 and conclusions are provided in Section 5.

2 FLOW SENSOR MODELING

2.1 The Basic Equations

In this work we have focused on the modeling of an anemometer type flow sensor as shown in Figure 1 [5].

For a liquid flow, the important types of thermal effects are heat conduction for a volume and heat convection for a surface. Therefore, for a parabolic profile of the horizontal fluid velocity $u(y)$, the basic steady state PDEs can be described by Equation (1) and the boundary conditions in Equation (2) [6]:

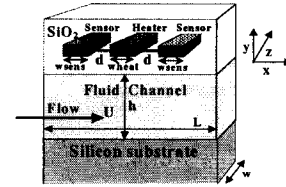


Figure 1. An anemometer type flow sensor.

Basic equations for $T(x, y)$:

$$\begin{aligned} \text{Silicon dioxide: } & k_{\text{SiO}_2} \text{div}(\nabla T) = 0 \\ \text{Substrate: } & k_{\text{Si}} \text{div}(\nabla T) = 0 \\ \text{Sensors: } & k_p \text{div}(\nabla T) + \chi_1 = 0 \\ \text{Heater: } & k_p \text{div}(\nabla T) + \chi_2 = 0 \\ \text{Fluid channel: } & k_{\text{fluid}} \text{div}(\nabla T) + \mu F(y) = \rho c_p u(y) \partial T / \partial x, \\ & F(y) = (\partial u / \partial y)^2, \quad u(y) = 4Uy(h-y)/h^2, \end{aligned} \quad (1)$$

Boundary conditions:

$$\begin{aligned} \text{Upper surface: } & k_{\text{SiO}_2} \partial T / \partial y = \eta(T - T_\infty) \\ \text{Left, right, and bottom edges: } & T(x=0) = T(x=L) = T(y=0) = T_\infty = \text{constant.} \end{aligned} \quad (2)$$

Here k_{Si} , k_{SiO_2} , k_p , k_{fluid} are the coefficients of thermal conductivity for silicon, silicon dioxide, polysilicon and the

¹ This work is sponsored in part by DARPA under agreement number F30602-98-2-0178.

fluid, respectively. χ_1 and χ_2 are the power corresponding to heat generation in the sensors and heater per unit volume, ρ is the fluid density, c_p is the thermal fluid capacity, η is the convection heat transfer coefficient, μ is the dynamic viscosity of the fluid, L is the length of the sensor, h is the height of the fluid channel, U is the maximum fluid velocity, and T_∞ is the ambient temperature.

The PDEs can be solved by using a numerical solver, such as CFD-ACE+ [7]. However, for circuit-level simulations a macromodel is required. The macromodel must be computationally efficient and provide an accurate description of the temperatures T_1 , T_2 , and T_3 for the upstream sensor, heater, and downstream sensor, respectively. Practical applications are based on the temperature difference $\Delta T = T_3 - T_1$ and on the heater temperature T_2 , so the model must be accurate for computing ΔT and T_2 .

2.2 The Modeling Approach

Our modeling approach uses a numerical solution of the PDEs for constructing the model. First a steady-state model is developed and this is then extended to include the dynamic behavior. The sequence of steps is outlined below:

- 1) A numerical solution is obtained for $T(x, y)$ from Equations (1) and (2) for different values of the velocity, U , the separation between sensors, d , the height of the fluid channel, h , and the sensor width, $wsens$
- 2) This solution yields discrete data points for T_1 , T_2 , and T_3 as a function of U , d , h , and $wsens$
- 3) The discrete data points are then approximated by neural networks

2.3 Numerical Solution of the PDEs

We have used the MATLAB PDE toolbox for obtaining a solution of the PDEs. Numerical simulation results are shown in Figures 2-3 for different values of the velocity U .

A parametric description of the sensor geometry, equations and boundary conditions is the basis for the model formulation. This description allows parametric studies, which are useful for the optimal design of the flow sensor. A post processing step is used to calculate temperatures T_1 , T_2 , and T_3 in the sensor sub-domains and to ensure that a zero value of ΔT is obtained for $U=0$.

2.4 The Design of Neural Networks

Neural Networks (NNs) have the following useful features [8]. They are:

- **General:** NNs produce reasonable outputs for inputs not encountered during the training (learning)
- **Robust:** NNs work well with noisy and incomplete data

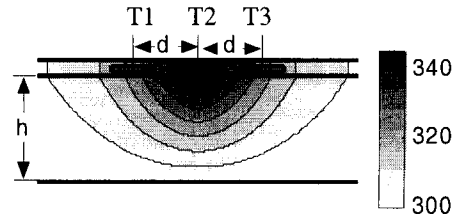


Figure 2. Solution of PDEs for $d=30 \mu\text{m}$, $h=20 \mu\text{m}$, $wsens=20 \mu\text{m}$, and $U=0$. For this case $T_1=320^\circ\text{K}$, $T_2=344.5^\circ\text{K}$, $T_3=320^\circ\text{K}$ and $\Delta T=0^\circ\text{K}$.

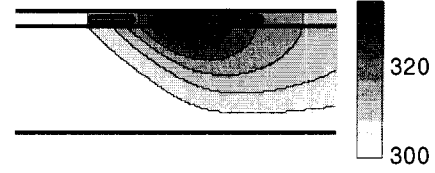


Figure 3. Solution of PDEs for $d=30 \mu\text{m}$, $h=20 \mu\text{m}$, $wsens=20 \mu\text{m}$, and $U=0.1 \text{ m/s}$. For this case $T_1=309^\circ\text{K}$, $T_2=334.1^\circ\text{K}$, $T_3=320.8^\circ\text{K}$, $\Delta T=11.8^\circ\text{K}$.

- **Universal:** Theoretically any smooth function can be approximated by NNs with an arbitrary precision

In real applications, the NNs are not an exact approximation, but a compromise between the accuracy of the approximation and other requirements. For flow sensor modeling, these requirements include a proper shape for the functions $\Delta T(U, d, h, wsens)$, $T_2(U, d, h, wsens)$, and $T_1(U, d, h, wsens)$.

The shape reconstruction function must be

- a smooth function of the arguments
- free from sharp transitions and noisy oscillations
- reasonably close to the physical behavior
- accurate for a large range of velocities

In our modeling approach, we have used a nonlinear scaling for U , h , and ΔT (because these values vary over a large interval) and a linear scaling for d , $wsens$, T_1 and T_2 . In addition, a new cost function is used for global optimization.

The nonlinear scaling is described by the transform $x' = (\log(x) + a)/b$ and the inverse transform $x = \exp(bx' - a)$, where a and b are chosen such that $x'_{\min} = -1$ and $x'_{\max} = 1$. A linear scaling yields $x'_{\min} = -1$ and $x'_{\max} = 1$. This scaling results in a

- better condition number for the Jacobian used in the Levenberg-Marquardt method [8]
- smoothing of the nonlinearity for improved optimization performance
- efficient stochastic search with random initialization of NN weights and biases

- the same optimum solution for both the absolute error of scaled ΔT and for the relative error of the original ΔT
- the same percentage error estimate for each scaled output value

The cost function for the global optimization is taken as a product: $f_{cost}=(N_c+1)*MSE$, where MSE is the mean squared error between the output data and outputs from the NNs, and N_c is the number of changes in the sign of the partial derivatives of the NN output with respect to the inputs. The minimum of this function yields both the shape reconstruction (a small number of the oscillations, expressed by N_c) and good accuracy (expressed by MSE). An optimum for f_{cost} avoids the over-fitting and under-fitting of data [8].

Because the proposed cost function is not smooth, stochastic search and genetic algorithms have been applied to solve this optimization problem.

3 MODEL VALIDATION

We have considered a microflow sensor with water as the fluid. The solution of PDEs from MATLAB were used to determine T1, T2, and T3 for different velocities in the range $U=[0, 10]$ m/s, separation in the range $d=[30, 200]$ μm and channel height in the range $h=[2, 200]$ μm . This data was used to construct the neural network.

An optimal NN model for ΔT was obtained as a NN with two hidden nonlinear layers and one linear output neuron. The results in Figures 4-7 show a good agreement between the model and the simulated data for a large range of U , d and h . The NN model can be used to predict ΔT for parameter values that were not included in the training set for the NN. An example of such a result is shown in Figure 7 where the model is compared with simulation data for a channel height of 50 μm . It should be noted that this data was not included in the training set and yet good agreement between the model and data is obtained.

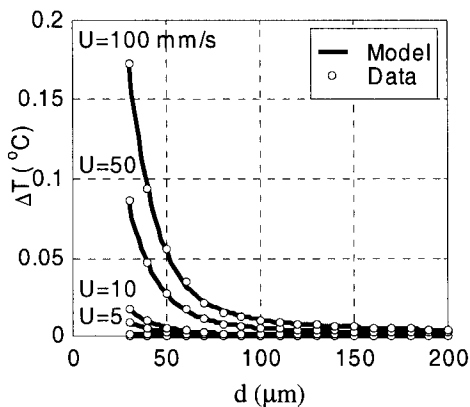


Figure 4. ΔT versus separation d for different velocities U with a channel height $h=2$ μm .

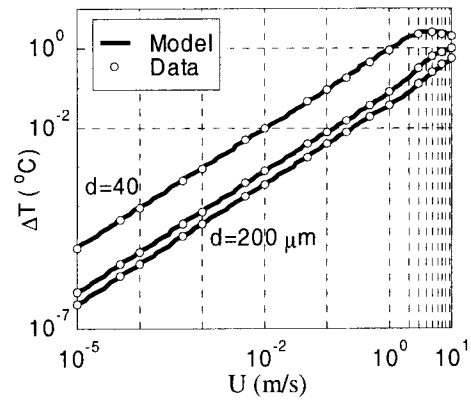


Figure 5. ΔT versus velocity U for different separations d with a channel height $h=2$ μm .

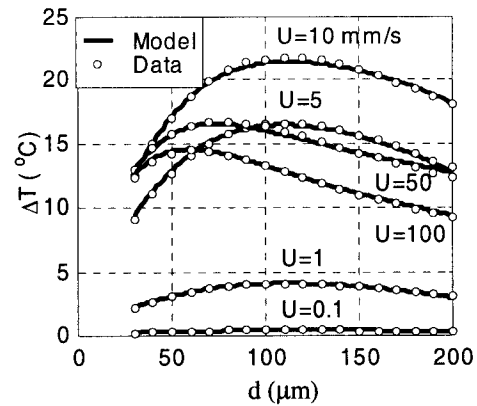


Figure 6. ΔT versus separation d for different velocities U with a channel height $h=100$ μm .

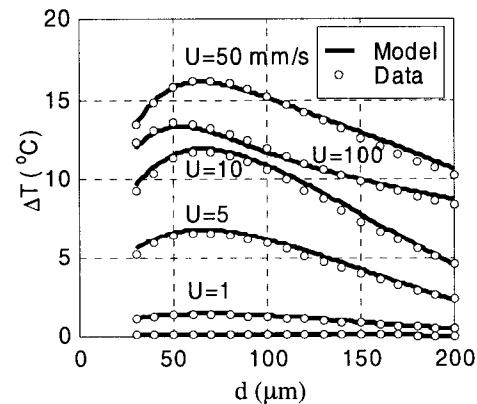


Figure 7. Predicted ΔT versus separation d for different velocities U with a channel height $h=50$ μm . The channel height of 50 μm was not used in the training set.

4 THE COMPLETE MODEL

In this section, we describe the complete steady-state model and the extension required for dynamic operation. The implementation of the model in a circuit simulator is also presented.

4.1 The Steady-State Model

The NN-based model was developed for a steady-state solution ΔT_{SS0} corresponding to a nominal power for the heat sources $\chi_1 = \chi_3 = \chi_{S0}$, and $\chi_2 = \chi_{h0}$. ΔT_{SS0} is a continuous function of the velocity, U , and the parameters, d , h , and w_{sens} . The steady-state ΔT (ΔT_{SS}) for a different power for the heat sources is obtained from this base model. Results from numerical simulations show that i) the heater power is the most important, and ii) ΔT_{SS} is a linear function of the power of the heat source. Therefore,

$$\Delta T_{SS} = \Delta T_{SS0} * \chi_{heat} / \chi_{heat0}$$

The macromodels for the T1 and T2 have the same form with different parameter values.

4.2 The Dynamic Model

The steady-state model for ΔT is extended for dynamic behavior, as shown in Figure 9.

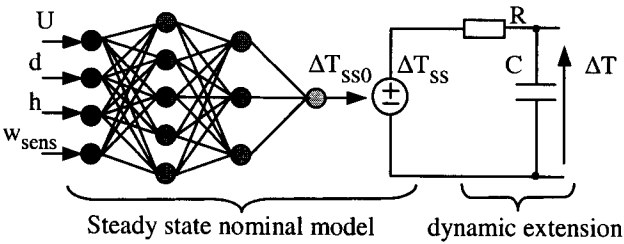


Figure 9. The dynamic macromodel for ΔT .

The dynamic behavior [9] is modeled by $\tau = RC = \rho D^2 / \mu$, $D = hw / (h + w)$, where ρ is the fluid density, μ is the dynamic viscosity of the fluid, h is the channel height, and w is the channel width. Because of a common fluid channel, the parameters for the dynamic extension are the same for ΔT , T1 and T2. T3 is calculated using $T3 = T1 + \Delta T$.

The complete macromodel for the flow sensor is composed of the individual macromodels as shown in Figure 10.

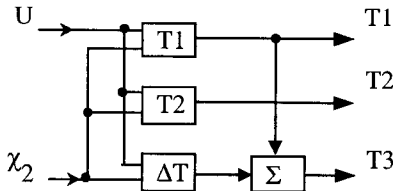


Figure 10. The complete flow sensor macromodel.

4.3 Model Implementation

The dynamic macromodel is incorporated in SPICE by coupling with a sensor circuit and a model for thermoresistors for the heater and sensors as shown in

Figure 11. Here, R1 and R3 are the resistances corresponding to the upstream and downstream sensors and R2 is the heater resistance. Based on the fluid flow rate T1, T2 and T3 change which in turn alters the resistance values and the sensing circuit currents and voltages [5].

This model can also be implemented in a VHDL-AMS simulator for the behavioral simulation of a microfluidic system.

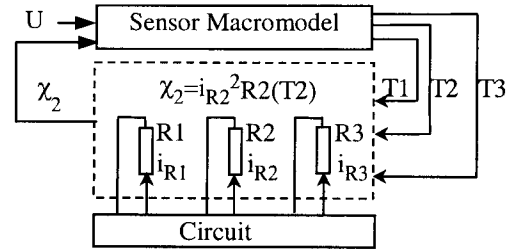


Figure 11. Macromodel implementation in SPICE.

5 CONCLUSION

An anemometer type microflow sensor macromodel has been developed using neural networks. The numerical solutions of PDEs describing the microflow sensor are used as training data for the neural networks. This model is simple and accurate and has been verified with numerical simulations for a wide range of sensor parameters. The model incorporates both the steady-state and dynamic responses of the flow sensor. The model has been implemented in SPICE3f5 and can be used for simulating the flow sensor in conjunction with an electronic circuit.

REFERENCES

- [1] N. R. Swart and A. Nathan, "Flow-rate microsensor modeling and optimization using SPICE," *Sensors and Actuators, A.*, 34, 109-122, 1992.
- [2] N. R. Swart and A. Nathan, "Mixed-mode device circuit simulation of thermal-based microsensors," *Sensors and Materials*, 6, 179-192, 1994.
- [3] S. Park, H. Kim, and Y. Kang, "Study of flow sensor using finite difference method," *Sensors and Materials*, 7, 43-51, 1995.
- [4] F. Mayer, G. Salis, J. Funk, O. Paul, and H. Baltes, "Scaling of thermal CMOS gas flow microsensors: experiment and simulation," *Proc. IEEE International Workshop on MEMS*, 126-121, 1996.
- [5] A. Rasmussen and M. E. Zaghoul, "The design and fabrication of microfluidic flow sensors," *Proc. ISCAS-99*, 136-139, 1999.
- [6] A. Rasmussen, et al., "Simulation and optimization of a microfluidic flow sensor," *In Preparation*.
- [7] <http://www.cfdrc.com>
- [8] R. D. Reed and R. J. Marks, "Neural smithing: supervised learning in feedforward artificial neural networks," *The MIT Press*, 1999.
- [9] J. Branebjerg, O. S. Jensen, N. G. Laursen, O. Leistiko, and H. Soeberg, "A micromachined flow sensor for measuring small liquid flows," *Proc. 1991 IEEE Transducers*, 41-44, 1991.