ELSEVIER

0141-0296(95)00156-5

*Engineering Structures*, Vol. 18, No. 11, pp. 856–860, 1996

0141-0296/96 $15.00 + 0.00

# Computation of flow around Texas Tech building using $k$-$\epsilon$ and Kato–Launder $k$-$\epsilon$ turbulence model

**R. Panneer Selvam**

*Department of Civil Engineering, BELL 4190, University of Arkansas, Fayetteville, AR 72701, USA*

An efficient model F3Dk$\varepsilon$ has been developed using a preconditioned conjugate gradient procedure to solve the pressure correction equation. It is eight times faster than the work presented by R.P. Selvam (*J. Wind Engng Indust. Aerodyn.* 1992, **41–44**, 1619–1627) and more than 16 times faster than explicit procedures. Using this code, the flow around the Texas Tech building is computed using the $k$-$\varepsilon$ and Kato–Launder $k$-$\varepsilon$ turbulence models. The computed pressures are compared with existing experimental results. The computed turbulent kinetic energy from the Kato–Launder $k$-$\varepsilon$ model is compared with that from the regular $k$-$\varepsilon$ model. Copyright © 1996 Elsevier Science Ltd.

**Keywords:** computational fluid dynamics, computational wind engineering, turbulence, building loads

## Introduction

In Refs [1,2] the pressure around the Texas Tech building is computed using the $k$-$\varepsilon$ turbulence model[3]. The computed pressures are in good agreement with field results except at the corners for certain angles of incidence. In Ref. [4], Murakami *et al.* compared the turbulent kinetic energy $k$ for different turbulence models around a bluff body and found that the $k$-$\varepsilon$ turbulence model over-estimates the $k$ in front of the building. Recently Kato–Launder in Ref. [5] suggested a modified $k$-$\varepsilon$ model. They mentioned that this model improved the prediction of $k$ compared to the standard $k$-$\varepsilon$ model. In this work the suggested $k$-$\varepsilon$ model in Ref. [5] is used to study the flow around the Texas Tech building. The computed pressure $p$ around the building is compared with available experimental results. The computed $k$ is compared with the $k$ from the standard $k$-$\varepsilon$ model.

In Ref. [1], an efficient solution procedure which is 60% faster than the usual procedure is suggested to solve the Navier–Stokes equations using the SIMPLE procedure. In this work a much faster solution procedure, the preconditioned conjugate gradient (PCG), is applied. This procedure yields solution about eight times faster than the one suggested in Ref. [1] in a scalar machine like a Sun workstation. Using this technique a computer model called F3DK$\varepsilon$ is developed. This computer model can predict wind flow around a building using $43 \times 36 \times 28$ cells in less than an hour in a Sun SPARCstation 10 or about 10 min in Cray Y-MP. The implementation of the PCG procedure will be discussed in detail for wind engineering problems.

*e*-mail: rps@engr.uark.edu

## Notation

| | |
|---|---|
| $k$ | turbulent kinetic energy |
| $p$ | mean pressure |
| **B, P, R, X, Z** | column vectors |
| **A** | symmetric matrix |
| $C_\mu$ | constant of 0.09 used in the $k$-$\varepsilon$ model |
| **D** | diagonal matrix |
| ICCG | incomplete Cholesky conjugate gradient |
| JCG | Jacobian conjugate gradient |
| **L** | lower triangular matrix |
| **M** | preconditioning matrix |
| MICCG | modified ICCG |
| PCG | preconditioned conjugate gradient |
| $P_k$ | production term |
| $U_i$ | mean velocity in the $x_i$ direction |
| $V$ | approach mean velocity |
| $\alpha,\beta$ | numbers computed in PCG |
| $\varepsilon$ | turbulent dissipation rate |
| $\gamma$ | constant ranging from 0 to 0.95 used in MICCG |
| $\rho$ | density of air |

## Computer modelling

*Turbulence modelling*

The flow around the building is modelled using the time averaged Navier–Stokes equations and the $k$-$\varepsilon$ turbulence model. The equations are given in detail in Ref. [1]. With respect to the $k$-$\varepsilon$ turbulence model; the standard $k$-$\varepsilon$ turbulence model (model 1) which was used in Ref. [1] and the new version used by Kato–Launder (model 2) in Ref. [5] are used in this work for comparison. The major differences between model 1 and model 2 are in the computation of the production term $P_k$. In model 1, there is an excessive generation of turbulence energy and $P_k$ in the vicinity of the stagnation point leads to far too high levels of turbulent viscosity, as reported in Refs [4,5]. To rectify this problem model 2 is suggested in Ref. [5]. In models 1 and 2, $P_k$ is computed as:

For model 1: $P_k = C_\mu \varepsilon S_1^2$ and for model 2: $P_k = C_\mu \varepsilon S_1 S_2$.

where

$$S_1 = \mathrm{SQRT}(0.5 \times (U_{ij}+U_{ji})^2) \times k/\varepsilon, C_\mu = 0.09\,,$$

where

$$S_2 = \mathrm{SQRT}(0.5 \times (U_{ij}-U_{ji})^2) \times k/\varepsilon\,.$$

Here subscripts of $U$ refer to direction in the tensor notation and a comma refers to differentiation. Using these two production terms, the pressures, turbulent kinetic energy and velocity around a building are computed for comparison.

*Numerical procedure*

The time averaged Navier–Stokes equations and the equations for the $k$-$\varepsilon$ turbulence model are solved in a rectangular, nonstaggered grid system as in Ref. [1]. The equations are integrated using the control-volume procedure. The momentum, $k$ and $\varepsilon$ equations are solved by SOR(successive over relaxation) point iteration with an under-relaxation parameter of 0.7 for each cycle. Usually 2–5 point iterations are sufficient for these equations at each cycle. To satisfy mass and momentum conservation simultaneously, the SIMPLE procedure[6] is used. Additionally, a pressure correction equation has to be solved for this. After the pressure correction the equations are solved, until that absolute sum of the residue reaches 0.3 times the initial value, the pressures are updated using an under relaxation factor of 0.2, as suggested in Ref. [1]. This cycle is repeated until the absolute sum of the residue of all the variables, except the pressure correction, reduces to 0.03 times the initial value. Each cycle is referred to as one outer iteration. The number of point iterations suggested for the momentum, $k$ and $\varepsilon$ equations and the convergence criteria for the pressure correction equations are the optimum values arrived at from numerical experimentation.

Usually the pressure correction equation is the most CPU time consuming step as reported in Ref. [7]. The rate of convergence of the pressure correction equation is very slow when SOR and other standard iterative procedures are used due to the Neuman boundary condition, and the way the coefficients of each equation are changing for different control volumes. In this work the PCG procedure is used to solve the pressure correction equations. This procedure is suitable for solving a symmetric set of simultaneous equations. The equations derived for a pressure correction step using the control volume procedure are symmetric and hence, the method is suitable.

As mentioned in Refs [8,9]; this procedure is more than 10 times faster than SOR. Also it has the advantage that there is no need to specify any acceleration parameters as in SOR. This procedure was already successfully used in Ref. [10] for computing the drag coefficient of the power conductors in Ref. [9] and the flow around a building due to a tornado. Using these techniques a computer program called F3Dk$\varepsilon$ has been developed.

*Implementation of PCG*

The following is the preconditioned conjugate gradient (PCG) algorithm for simultaneous equations of the type $\mathbf{AX} = \mathbf{B}$, where $\mathbf{A}$ is a symmetric matrix and $\mathbf{X}$ and $\mathbf{B}$ are unknown and known vectors.

*PCG Algorithm.*

Start from: $\mathbf{R}_1 = \mathbf{B}-\mathbf{AX}_1$

for $i = 1,2,...$

$\mathbf{Z}_i = \mathbf{M}^{-1}\mathbf{R}_i$

if $i = 1$

$\mathbf{P}_1 = \mathbf{Z}_1$

else

$\beta = \mathbf{Z}_i^{\mathrm{t}}\mathbf{R}_i\mathbf{Z}_{i-1}\mathbf{R}_{i-1}$

$\mathbf{P}_i = \mathbf{Z}_i+\beta\mathbf{P}_{i-1}$

end if

$\alpha = \mathbf{Z}_i^{\mathrm{t}}\mathbf{R}_i/\mathbf{P}_i^{\mathrm{t}}\mathbf{A}\mathbf{P}_i$

$\mathbf{X}_{i+1} = \mathbf{X}_i+\alpha\mathbf{P}_i$

$\mathbf{R}_{i+1} = \mathbf{R}_i-\alpha\mathbf{A}\mathbf{P}_i$

if $\mathbf{R}_i = 0$ or reached the required convergence then stop or go for next iteration $i$

In this $\mathbf{B}$, $\mathbf{P}$, $\mathbf{R}$, $\mathbf{X}$ and $\mathbf{Z}$ are vectors and $\alpha$ and $\beta$ are scalars. Here $\mathbf{M}$ is the preconditioning matrix and superscript t refers to transpose of a matrix. A simple choice will be a diagonal matrix whose diagonal elements will be the same as those of $\mathbf{A}$. This preconditioner is called the Jacobian conjugate gradient (JCG). Many other versions of $\mathbf{M}$ are given in Refs [11,12]. The preferred one with respect to storage limitation and efficiency in implementation will be the incomplete Cholesky conjugate gradient (ICCG) in Ref. [12]. In that, the $\mathbf{M}$ matrix is an incomplete Cholesky factorization of the matrix $\mathbf{A}$ without any fill in. $\mathbf{M}$ can be written as $(\mathbf{L}+\mathbf{D})\mathbf{D}^{-1}(\mathbf{D}+\mathbf{L})^{\mathrm{t}}$. $\mathbf{L}$ is a strict lower triangular matrix whose elements are same as $\mathbf{A}$, and $\mathbf{D}$ is a diagonal matrix. Using the notation of ae, aw, an, as, at, ab and ap for east, west, north, south, top, bottom and diagonal coefficients, respectively, in a control volume procedure for a three-dimensional rectangular grid as in Ref. [6]; $\mathbf{D}$ can be written as follows:

$$\mathbf{D}_{i,j,k} = \mathrm{ap}_{i,j,k}-t1-t2-t3$$

where

$$t1 = \mathrm{ae}_{i-1,j,k}*(\mathrm{ae}_{i-1,j,k}+\gamma*(\mathrm{an}_{i-1,j,k}+\mathrm{at}_{i-1,j,k}))/\mathrm{d}_{i-1,j,k}$$

$$t2 = \mathrm{an}_{i,j-1,k}*(\mathrm{an}_{i,j-1,k}+\gamma*(\mathrm{ae}_{i,j-1,k}+\mathrm{at}_{i,j-1,k}))/\mathrm{d}_{i,j-1,k}$$

$$t3 = \mathrm{at}_{i,j,k-1}*(\mathrm{at}_{i,j,k-1}+\gamma*(\mathrm{ae}_{i,j,k-1}+\mathrm{an}_{i,j,k-1}))/\mathrm{d}_{i,j,k-1}\,.$$

Here the indices $i$, $j$ and $k$ vary for a number of points in the $x$,$y$ and $z$ directions. Because of symmetry, only the upper or lower triangular part is stored. If the $\mathbf{L}$ matrix is stored, it will have aw,as and ab, and if $\mathbf{L}^{\mathrm{t}}$ is stored it will

have ae, an and at. Once **D** matrix is calculated; the solution of **MZ** = **R** is by regular forward and backward substitution. To implement this procedure only 11 vector storage spaces are needed for ae, an, at, ap, **D**, **B**, **X**, **R**, **Z**, **P** and ap. The solution of **MZ** = **R** is mostly scalar computation and the rest of the PCG algorithm is highly vectorizable. Methods to vectorize the solution process of **MZ** = **R** are discussed in Refs [12,13].

Usually $\gamma$ is taken as being close to 1, but less than 1 in calculating the **D** matrix. This implementation is called the modified ICCG, i.e. MICCG(1,1). If $\gamma = 0$, then the procedure is called (ICCG(1,1). In this work a $\gamma$ value of 0.95 as recommended by Van Der Vorst in Ref. [13] is used. For the first outer iteration the pressure correction solution with the ratio of final absolute sum of residue to initial absolute sum of residue is 0.3; the ICCG took 35 iterations and the MICCG with $\gamma = 0.95$ took 22 iterations. Since both procedures have the same amount of computational work once **D** is calculated, there is a decrease in computation by 37% for MICCG compared with ICCG. The number of iterations needed to solve the pressure correction equation is less than 10 after the first outer iteration.

*Computational efficiency of F3Dkε*

For comparison of computer time with the work in Ref. [1]; the same grid size of $43 \times 36 \times 28$ in Ref. [1] is considered. The F3Dkε using the $k$-$\varepsilon$ model 1; took 58 iterations to converge and 170 min in Sun SPARCstation IPC, 49 min in Sun SPARCstation 10 and 8 min in Cray-Y MP which has a maximum performance of 330 Mflobs. Compared to this the line iteration work in Ref. [1] took about eight times the CPU time of F3Dkε in a scalar machine like the sun workstation.

The F3Dkε is not completely vectorized. Even though the JCG procedure is completely vectorizable and could achieve a performance of 100 Mflobs; the over-all program could achieve only 17.9 Mflobs. The MICCG procedure could achieve 42.4 Mflobs and the overall code achieved 13.86 Mflobs. With respect to CPU time; the MICCG took less than the JCG. If the methods suggested in Ref. [13] are utilized to further vectorize the MICCG procedure, the MICCG may be much faster than the JCG.

When the F3Dkε uses a steady-state solution procedure; codes like TEMPEST, EXACT3, and Murkami *et al.* in Ref. [4] use an explicit form of calculation as reported in Ref. [14]. In comparison to 8 min of CPU time in Cray Y-MP using the F3Dkε; the TEMPEST took about 2.23 hr of CPU time in Cray Y-MP[14] and Murakami and his group took more than 5 hr of CPU time in the equivalent machine as reported in Ref. [15] to reach a steady-state solution. Hence unless time dependent phenomena are necessary it is cheaper to use steady-state or implicit procedures which lead to steady-state.

## Results

The Texas Tech experimental building has the dimension of $9.1 \times 13.7 \times 4$ m as reported in Ref.[16]. The building and the flow region is discretized by $43 \times 36 \times 28$ points. The grid variations along the $xz$ and the $yz$ plane are shown in *Figures 1* and *2*. The parameters considered for the wind flow far away from the building are the same as those of Ref. [1]. The wind is assumed to flow along the 9.1 m width of the building. The pressure coefficients are calculated as

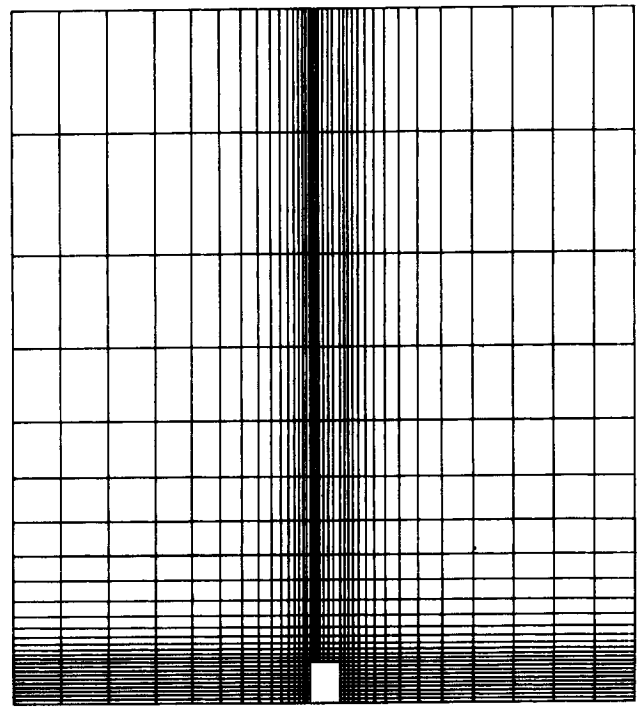

*Figure 1* Discretization in the $xz$ plane

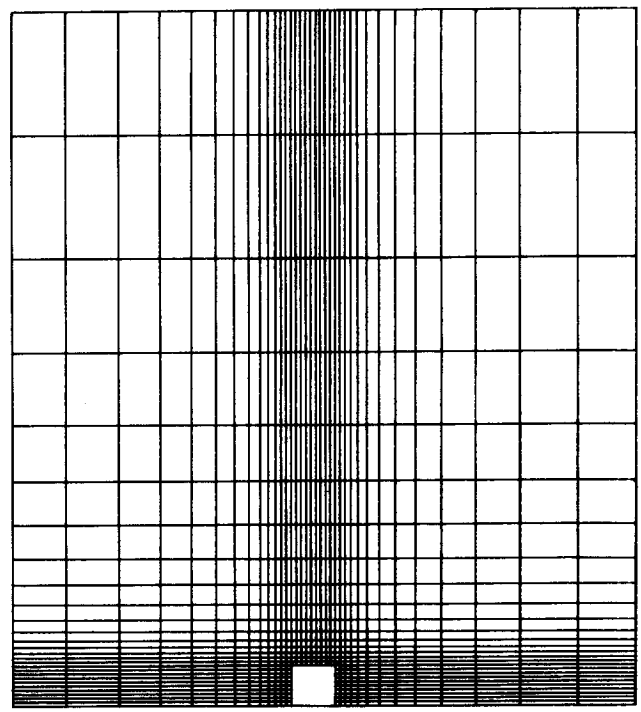

*Figure 2* Discretization in the $yz$ plane

$$Cp = 2(p - p_0)/(\rho V^2),$$

where $p$ is the mean pressure, $\rho$ is the density, $V$ is the approach mean velocity at building height and $p_0$ is the reference mean pressure at free flow considered to be zero here.

Using the F3Dkε, flow around the building is computed using the regular $k$-$\varepsilon$ and Kato–Launder $k$-$\varepsilon$. The Kato-Launder $k$-$\varepsilon$ model took 66 iterations and 87 min compared with 58 iterations and 49 min by a Sun SPARCstation 10 using a regular $k$-$\varepsilon$ model. The computed pressures using the regular and Kato–Launder $k$-$\varepsilon$ model are compared

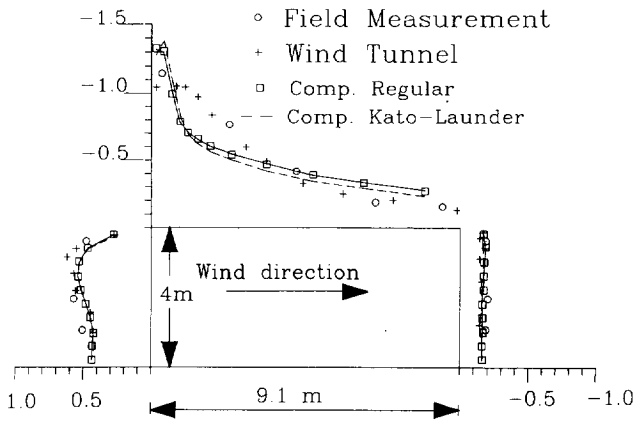


*Figure 3* Comparison of mean pressure coefficients

with the available field[16] and wind tunnel[17] results in *Figure 3*. The computed pressures from both turbulence models are almost the same except at the leeward roof. At the leeward roof the Kato–Launder predictions are much closer to the field and wind tunnel results.

The computed velocity field from the regular and modified $k$-$\varepsilon$ model at the same plane where the pressures are compared is shown in *Figures 4* and *5*. The velocity vector diagrams from both models are almost identical. In *Figures 6* and *7* the contours of turbulent kinetic energy for the regular and Kato–Launder $k$-$\varepsilon$ model are plotted for the same center plane. The regular $k$-$\varepsilon$ model has a maximum of 5 on the windward side of the building. Whereas in the case of the Kato–Launder $k$-$\varepsilon$ model, the maximum appears to be about 2.5. The Kato–Launder model predictions of the turbulent kinetic energy are much closer to the real situation. Since no measurements were available for $k$; it was not possible to compare the computed value with field or wind-tunnel measurements.

## Conclusions

An efficient computer model F3Dk$\varepsilon$ using the modified incomplete Cholesky conjugate gradient (MICCG) procedure to solve the pressure correction is developed. Most of MICCG procedure is highly vectorizable. The algorithm to implement MICCG for the wind engineering problem is discussed in detail. The MICCG procedure is about eight times faster than the line iteration work reported in Ref. [1]. The F3Dk$\varepsilon$ took only about 49 min in a Sun SPARCstation 10 to compute the flow on a $43 \times 36 \times 28$ grid. In Cray Y-MP when the F3Dk$\varepsilon$ took 8 min, codes based on explicit unsteady solution technique like the TEMPEST in Ref. [14] took 2.23 hr and Murakami and his group[15] in an equivalent machine took more than 5 hr to reach steady state. Hence to compute steady-state velocity field the F3Dk$\varepsilon$ is very efficient and suitable to run in a workstation environment.

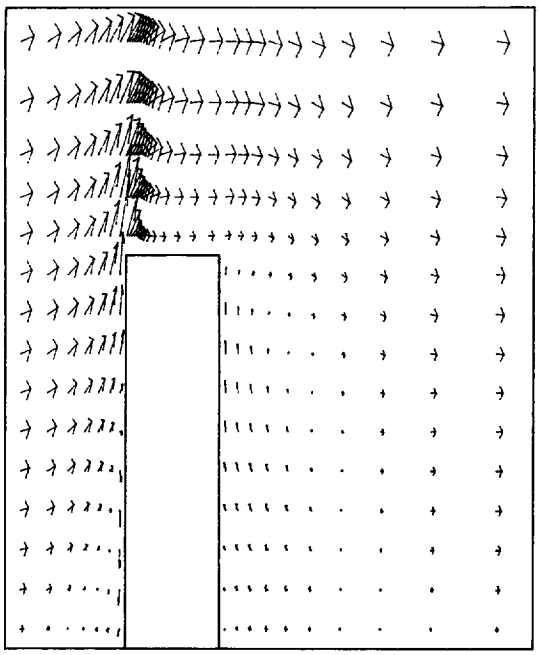

*Figure 4* Velocity vector diagram for regular $k$-$\varepsilon$ model

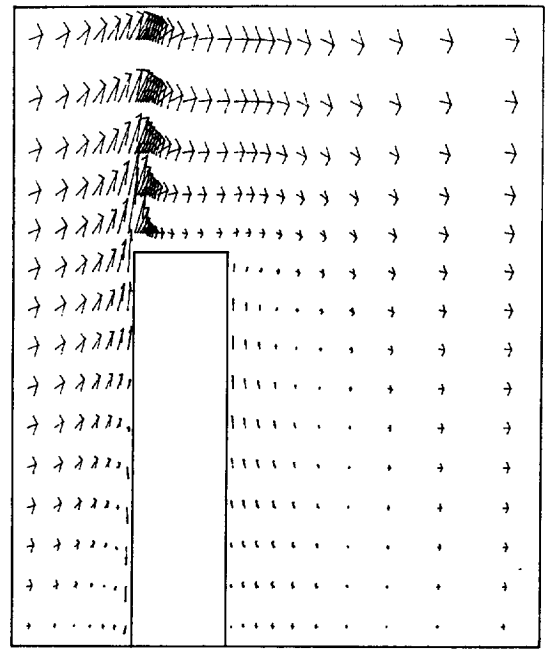

*Figure 5* Velocity vector diagram for Kato–Launder $k$-$\varepsilon$ model

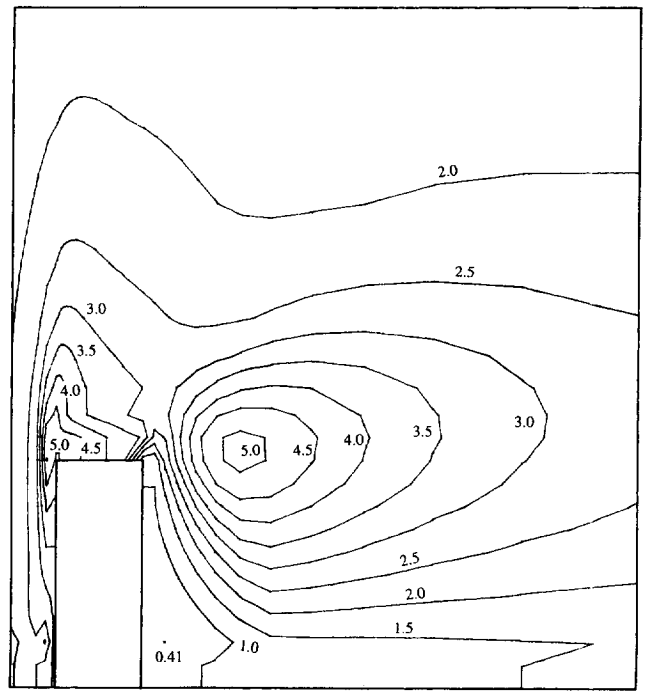


*Figure 6* Turbulent kinetic energy contour for regular $k$-$\varepsilon$ model

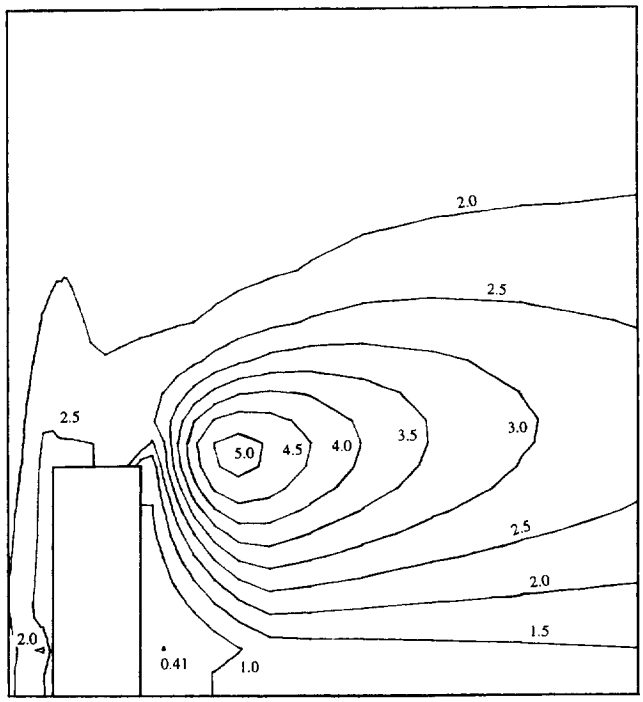


*Figure 7* Turbulent kinetic energy contour for Kato–Launder $k$-$\varepsilon$ model

The computed pressure coefficients on the Texas Tech experimental building are compared with available experimental results. The computed pressure coefficients and velocity field over the center plane of the building from both models are almost the same. The computed turbulent kinetic energy from the Kato–Launder model is much lower than the regular $k$-$\varepsilon$ model on the windward side of the building. Hence, the Kato–Launder model predictions are much closer to reality. It can be concluded from this work that if one is interested in velocity or pressure around a building, it seems that it is immaterial which procedure is used.

## References


1 Selvam, R. P. Computation of pressures on Texas Tech Building, *J. Wind Engng Ind. Aerodyn.* 1992, **43**, 1619–1627
2 Selvam, R. P. and Konduru, P. B. Computational and experimental roof corner pressures on the Texas Tech Building, *J. Wind Engng Ind. Aerodyn.* 1993, **46/47**, 449–454
3 Launder, B. E. and Spalding, D. B. The numerical computation of turbulent flows, *Comput. Meth. Appl. Mech. Engng* 1974, **3**, 269–289
4 Murakami, S., Mochida, A., Hayashi, Y. and Sakamoto, S. Numerical study on velocity-pressure field and wind forces for bluff bodies by $k$-$\varepsilon$, ASM and LES, *J. Wind. Engng Ind. Aerodyn.* 1992, **44**, 2841–2852.
5 Kato, M. and Launder, B. E. The modelling of turbulent flow around stationary and vibrating square cylinders, *Ninth Symp. on Turbulent Shear Flows*, Kyoto, Japan, 16–18 August 1993, Vol. 10–4, pp. 1–6
6 Patankar, S. V. *Numerical heat transfer*, McGraw Hill, New York, 1980
7 Van Doormal, J. P. and Raithby, G. D. Enhancements of the SIMPLE method for predicting incompressible fluid flows, *Numer. Heat Transfer* 1984, **7**, 147–163
8 Kershaw, D. S. The incomplete Cholesky-conjugate gradient method for the iterative solution of systems of linear equations, *J. Comput Phys.* 1978, **26**, 43–65
9 Selvam, R. P. and Paterson, D. A. Computation of conductor drag coefficients, *J. Wind Engng Ind. Aerodyn.* 1993, **50**, 1–8
10 Selvam, R. P. Computer modelling of tornado forces on buildings, *Proc. 7th US National Conf. on Wind Engineering*, 27–30 June, 1993, Vol. II, pp. 605–613.
11 Meijerink, J. A. and Van der Vorst, H. A. Guidelines for the usage of incomplete decompositions in solving sets of linear equations as they occur in practical problems. *J. Comput. Phys.* 1981, **44**, 134–155
12 Ashcraft, C. C. and Grimes, R. G. On vectorizing incomplete factorization and SSOR preconditioners, *SIAM J. Sci. Stat. Comput.* 1988, **9**, 122–151
13 Van der Vorst, H. A. High performance preconditioning, *SIAM J Sci. Stat. Comput.* 1989, **10** 1174–1185
14 Selvam, R. P. and Huber, A. H. Computer modelling of pollutant dispersion around buildings: current status. Report of Department of Civil Engineering, University of Arkansas, Fayetteville, AR, August 1993
15 Stathopoulos, T. and Kind, R. J. Discussion of computational fluid dynamics, *J. Wind Engng Ind. Aerodyn.* 1992, **44**, 2875–2879
16 Levitan, M. L., Holmes, J. D., Mehta, K. C. and Vann, W. P. Field measurement of pressures on the Texas Tech Building, *J. Wind Engng Aerodyn.* 1991, **38**, 227–234
17 Surry, D. Pressure measurements on the Texas Tech Building: wind tunnel measurements and comparison with full scale, *J. Wind Engng Ind. Aerodyn.* 1991, **38**, 235–247