1988

Boundary Layer on a Continuous Flat Plate in Parallel Flow with Similar and Non-Similar Boundary Conditions

Vaughn H.M. Faber
University of Central Florida

BOUNDARY LAYER ON A CONTINUOUS FLAT PLATE IN PARALLEL FLOW WITH SIMILAR AND NON-SIMILAR BOUNDARY CONDITIONS

BY

VAUGHN H. M. FABER
B.S.M.A.E., Illinois Institute of Technology, 1980

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Engineering
in the Graduate Studies Program
of the College of Engineering
University of Central Florida
Orlando, Florida

Spring Term
1988

ABSTRACT

This thesis is a computational analysis of the incompressible laminar boundary layer on a continuous moving surface. The skin friction and heat transfer variations on a moving plate with similar and non-similar (constant) mass transfer at the wall are obtained for 210 cases for the similar boundary solution and 60 cases for the non-similar solution.

Both cases were solved numerically using the fourth order Runge-Kutta algorithm. For the non-similarity case, the local non-similarity method as described by Sparrow, Quack and Boerner was used.

The analysis shows that the shear stress for both the similar and non-similar cases is higher when the velocity of the wall is greater than the free stream velocity. It decays with increasing mass transfer at the wall. The shear stress also increases as the velocity difference is enlarged.

Analysis was also made to determine the heat transfer from the plate for the similar boundary layer cases. The results of the analysis show that heat transfer is greatest when the wall velocity equals the free stream velocity.

Also, heat transfer was enhanced by increasing the injection parameter and decays with increasing velocity difference.

Practical applications of this flow field are extrusion of porous plastic sheets, quenching of large perforated metallic sheets, and water condensing on aircraft during flight. Water condensation during flight is of interest in the problem of aircraft icing.

ACKNOWLEDGEMENTS

The preparation of this thesis required the assistance of many people. The number of people who gave me assistance, both great and small, was greatly appreciated; however, space does not permit listing all of them.

I would like to give special thanks to my advisor, Dr. Loren Anderson, whose many suggestions and guidance made this effort possible. I would also like to thank Padmanabha Chappidi for his valuable assistance and many helpful suggestions. Lastly, but by no means least, I would like to thank my wife, Loraine. Without her constant help and encouragement this thesis, much less graduate school, would not have been possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

C = dimensionless injection parameter for similar problem = $v_w(\nu U_r/x)^{-1/2}$

f = function which is proportional to Ψ

p = the derivative of f with respect to η

q = the derivative of p with respect to η

g = derivative of f with respect to ζ

r = derivative of g with respect to η

s = derivative of r with respect to η

Nu = Nusselt number

Pr = Prandtl number

Re = Reynolds number

t = derivative of θ with respect to η (in Runge-Kutta formulas)

T = temperature

u = local velocity parallel to the plate

u = derivative of Φ with respect to η (in Runge-Kutta formulas)

U = velocity parallel to the plate

v = local velocity perpendicular to the plate

$\Delta_n X_i$: n = 1,2,3,4; X = f, g, p, q, r, s, θ, Φ, t, u: = Runge-Kutta difference values

ζ = dimensionless injection parameter for non-similar problem = $(v_w/U_r)(2U_r x/\nu)^{1/2}$

η = dimensionless coordinate perpendicular to plate

= $y(U_r/\nu x)^{1/2}$ (similar case)

= $y(U_r/2\nu x)^{1/2}$ (non-similar case)

θ = dimensionless temperature = $(T - T_\infty)/(T_w - T_\infty)$

ν = kinematic viscosity

Φ = derivative of θ with respect to ζ

Ψ = stream function

= $(\nu U_r x)^{1/2} f(\eta)$ (similar case)

= $(2\nu U_r x)^{1/2} f(\zeta,\eta)$ (non-similar case)

Subscripts

i = index for cell indices

r = reference

w = wall condition

x,y = differentiation with respect to x or y respectively

ζ = differentiation with respect to ζ

∞ = free stream condition

INTRODUCTION

This thesis investigates the boundary layer on a moving continuous flat plate with similar and non-similar mass transfer. The first part of this thesis investigates the case of similar mass transfer and the second part deals with non-similar mass transfer. For purposes of this thesis, similar mass transfer is when the injection velocity at the wall is proportional to the reciprocal of the square root of x, where x is a characteristic dimension along the plate. Constant mass transfer is the non-similar case. For the similar and non-similar cases, velocity profiles are computed. For the similar case, temperature profiles are determined; however, in the non-similar case computational problems prevent temperature profiles from being obtained.

The continuous moving flat plate (CF) problem, shown in Figure 1b, is different from the Blasius flat plate problem (BF), shown in Figure 1a, as stated by Abdelhafez [1]. In the continuous moving flat plate (CF) problem the coordinate system is fixed in space, whereas in the Blasius flat plate (BF) problem the coordinate system is fixed on the plate. This relationship remains the same for the continuous moving flat plate (CF) problem and Blasius flat plate problem when boundary layer injection or suction is included.

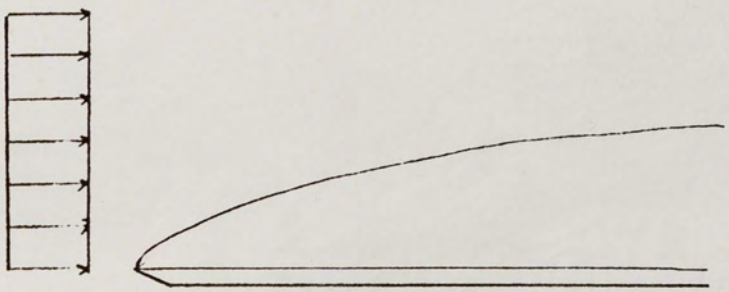

Figure 1a. Boundary Layer for Flow on a Flat Plate.

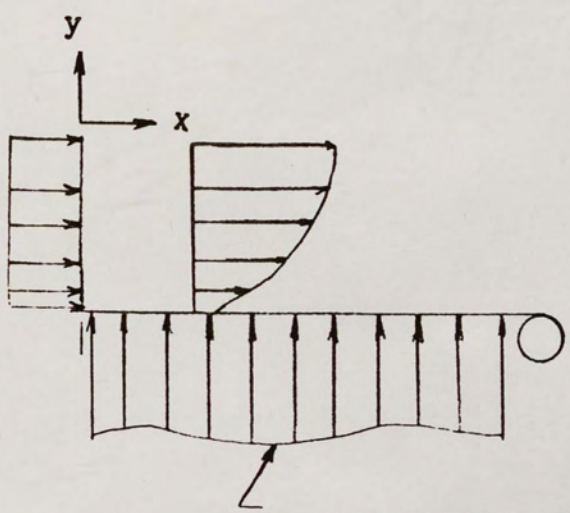

Figure 1b. Boundary Layer for Flow on a Moving Wall.

In the continuous moving flat plate (CF) problem, inclusion of blowing or suction at the plate does not affect the tendency for both the skin friction and heat transfer to be greater for cases where $U_w > U_\infty$ with the same velocity difference, $|U_w - U_\infty|$, and surface mass transfer. This is due to the change in direction of the transverse velocity component [1]. The differing values for skin friction and heat transfer between the continuous moving flat plate (CF) problem with injection or suction (CFI) and the Blasius flat plate (BF) problem with injection or suction (BFI) is an example that the two flows are different and cannot be mathematically transformed into one another with $|U_w - U_\infty|$ as a parameter using a Galilean transformation. A Galilean transformation is when components of a given vector in two different coordinate systems are related to each other by an orthogonal transformation [2].

ANALYSIS

The first part of this thesis explores the boundary layer for the CF problem with similar injection (the injection velocity $v_w = C(\nu U_r/x)^{1/2}$). The CFI problem is steady for a coordinate system fixed in space.

The second part of this thesis attempts to solve the CF problem with constant boundary layer injection at the plate surface. This problem is also known as non-similar boundary layer injection.

Similar Boundary Layer Case

As is usual for problems of this type, dimensionless variables are employed to make the results fit a broader range of applications and make some simplifications in the problem.

The following boundary layer equations can be derived by use of a rectangular control volume (subscripts denote differentiation with respect to the indicated variable):

$$u_x + v_y = 0 \qquad \text{(continuity)} \qquad (1)$$

$$uu_x + vu_y = \nu u_{yy} \qquad \text{(momentum)} \qquad (2)$$

and

$$uT_x + vT_y = (T_{yy})/Pr \qquad \text{(energy)} \qquad (3)$$

with the boundary conditions

$$@\ y = 0;\ u = U_w,\ v = v_w,\ T = T_w$$

where

$$v_w = C(\nu U_r/x)^{1/2}.$$

For this problem a dimensionless variable, η, is defined as

$$\eta = y(U_r/\nu x)^{1/2}.$$

From elementary fluid mechanics it is known that the stream function, Ψ, satisfies the continuity equation. The stream function for this problem, as in the BF problem, is defined as

$$\Psi = (U_r \nu x)^{1/2} f(\eta)$$

where U_r is the reference velocity and is defined as

$$U_r = U_\infty \text{ if } U_\infty > U_w$$

$$U_r = U_w \text{ if } U_w > U_\infty.$$

Also for a potential flow it can be shown that

$$u = \Psi_y \quad \text{and} \quad v = -\Psi_x.$$

The solution to equation (2) with these variables is given by Schlichting [3] as

$$f''' + (1/2)ff'' = 0 \tag{4}$$

and the transformed boundary conditions become

$$@\ \eta = 0;\ f = -2C \tag{5a}$$

$$f' = U_w/U_r \tag{5b}$$

$$@\ \eta = \infty;\ f' = U_\infty/U_r. \tag{5c}$$

Next, to solve equation (3), a dimensionless temperature, θ, is defined as

$$\theta = (T - T_\infty)/(T_w - T_\infty).$$

According to Burmeister [4], this transforms equation (3) to

$$\theta'' + (1/2)Pr f\theta' = 0. \tag{6}$$

The boundary conditions become

$$\theta(0) = 1.0 \tag{7a}$$

$$\theta(\infty) = 0.0. \tag{7b}$$

Equation (4) with (5a), (5b) and (5c) are then solved numerically by the fourth order Runge-Kutta method. Chow [5] gives, for solving equation (4), the equations to be used in the fourth order Runge-Kutta methods as

$$f' = p \tag{8a}$$

$$f'' = p' = q \tag{8b}$$

$$f''' = q' = -(1/2)fq. \tag{8c}$$

After applying the standard Runge-Kutta formulas, the following difference equations are obtained:

$$\Delta_1 f_i = hp_i \tag{9a}$$

$$\Delta_1 p_i = hq_i \tag{9b}$$

$$\Delta_1 q_i = -.5hf_i q_i \tag{9c}$$

$$\Delta_2 f_i = h(P_i + .5\Delta_1 p_i) \tag{9d}$$

$$\Delta_2 p_i = h(q_i + .5\Delta_1 q_i) \tag{9e}$$

$$\Delta_2 q_i = -.5h(f_i + .5\Delta_1 f_i)(q_i + .5\Delta_1 q_i) \tag{9f}$$

$$\Delta_3 f_i = h(p_i + .5\Delta_2 p_i) \tag{9g}$$

$$\Delta_3 p_i = h(q_i + .5\Delta_2 q_i) \tag{9h}$$

$$\Delta_3 q_i = -.5h(f_i + .5\Delta_2 f_i)(q_i + .5\Delta_2 q_i) \tag{9i}$$

$$\Delta_4 f_i = h(p_i + \Delta_3 p_i) \tag{9j}$$

$$\Delta_4 p_i = h(q_i + \Delta_3 q_i) \tag{9k}$$

$$\Delta_4 q_i = -.5h(f_i + \Delta_3 f_i)(q_i + \Delta_3 q_i). \quad (91)$$

These values are then used to compute f, p, and q at i + 1 as follows:

$$f_{i+1} = f_i + (\Delta_1 f_i + 2(\Delta_2 f_i + \Delta_3 f_i) + \Delta_4 f_i)/6 \quad (10a)$$

$$p_{i+1} = p_i + (\Delta_1 p_i + 2(\Delta_2 p_i + \Delta_3 p_i) + \Delta_4 p_i)/6 \quad (10b)$$

$$q_{i+1} = q_i + (\Delta_1 q_i + 2(\Delta_2 q_i + \Delta_3 q_i) + \Delta_4 q_i)/6. \quad (10c)$$

Since a boundary condition for f''(0) is not given, a value is assumed for f''(0) and the integration is carried out until etamax is reached. The value of f'(etamax) is compared to the value given for f'(∞) in equation (5c). For this problem etamax is 10. If f'(etamax) is within ε of f'(∞), then the assumed value of f''(0) is taken as correct and the integration is stopped. If f'(etamax) is not within ε of f'(∞), then a new value of f''(0) is assumed. In order to make more efficient use of the computer, a new value for f''(0) is chosen by use of the secant method. Zucrow [6] explains the secant method which is a numerical technique that significantly accelerates convergence in iterative calculations.

Once the solution of equation (4) is obtained, the solution to equation (6) can be solved. For equation (6) a different method is taken. Chow [5] derives a set of difference equations which are solved simultaneously to obtain the solution to a general second order ordinary differential equation. This method is executed in lines 77

through 100 and subroutine TRID in the program listing in Appendix A, with a detailed description given by Chow [5].

The result of this analysis is shown in Figures 2 - 16. Figures 2 - 5, 7 - 10 and 12 - 15 show the velocity and temperature profiles for various values of the similarity parameter (C) and velocity differences $|U_w - U_\infty|$. From these figures it is seen that the velocity profiles for the cases where $U_\infty > U_w$ are similar to those with $U_w > U_\infty$ for the same injection parameter (though rotated).

As shown in Figures 2 - 5, the boundary layer thickens as the injection parameter (C) is increased and as the velocity difference is increased. The reason for this behavior is that energy is added with increasing injection parameter (C) and this results in a more gradual velocity profile. Additionally, as the velocity difference between the free stream and plate increases, there is a greater velocity gap to span, thus resulting in a thicker boundary layer.

Figure 6 shows how the friction coefficient varies with both $U_w - U_\infty$ and the injection parameter (C). Figure 6 also shows that the friction coefficient (C_f), is greater when $U_w > U_\infty$ for comparable values of C than when $U_\infty > U_w$. This is, as stated earlier, the result of the change in direction of the transverse velocity vector.

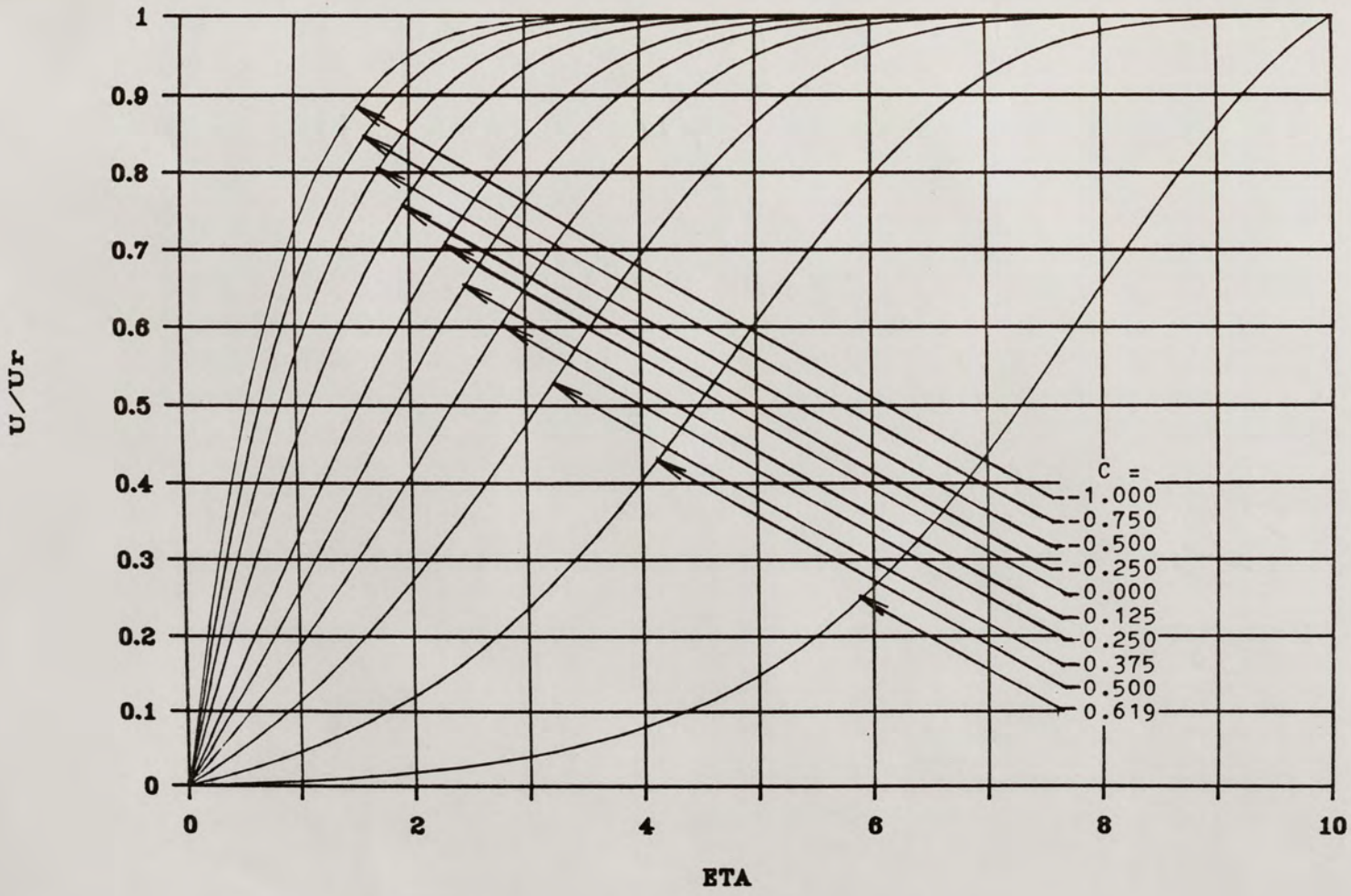

Figure 2. Velocity Profiles for $U_\infty - U_W = 1.$

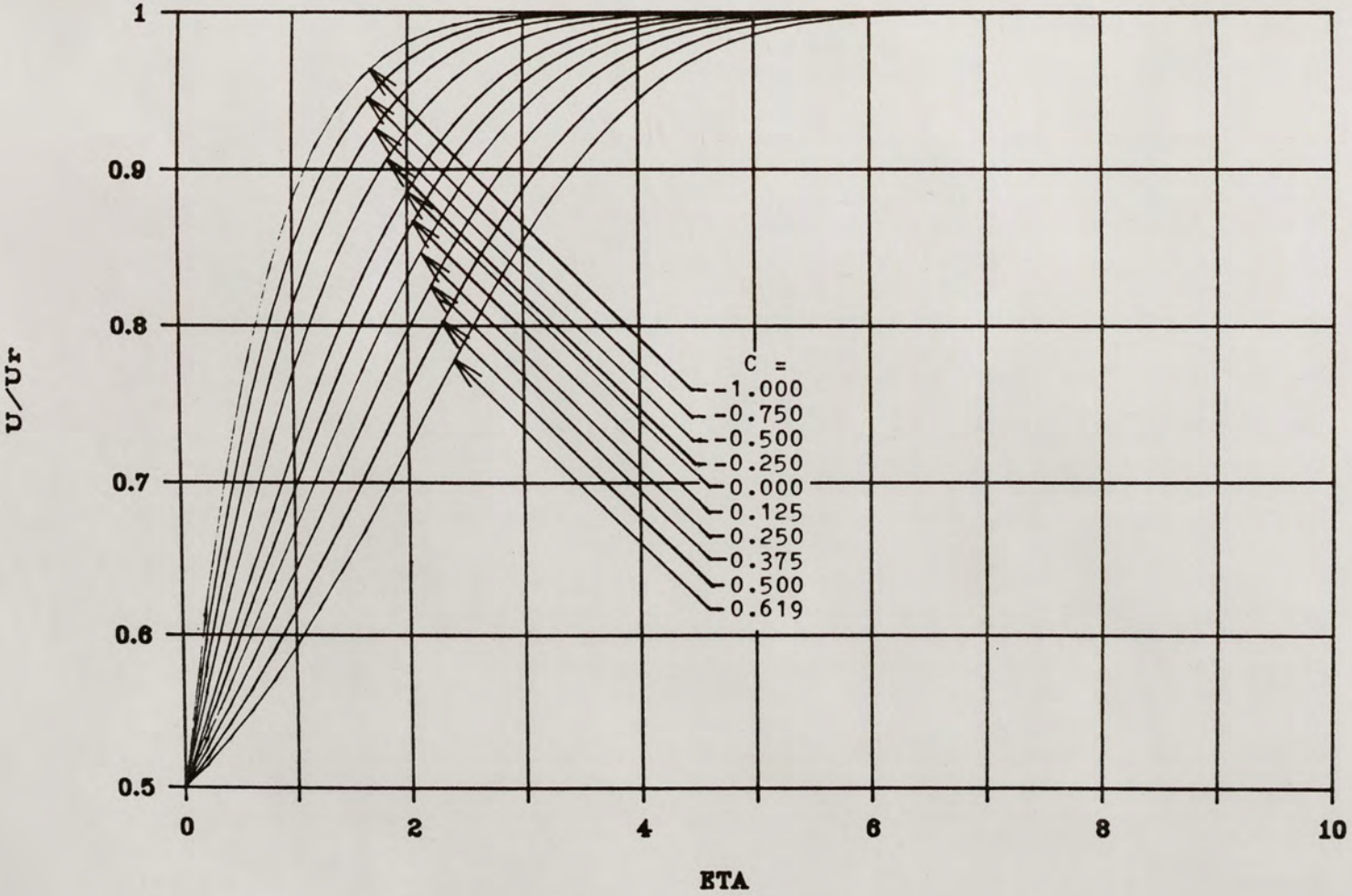

Figure 3. Velocity Profiles for $U_\infty - U_w = 0.5$.

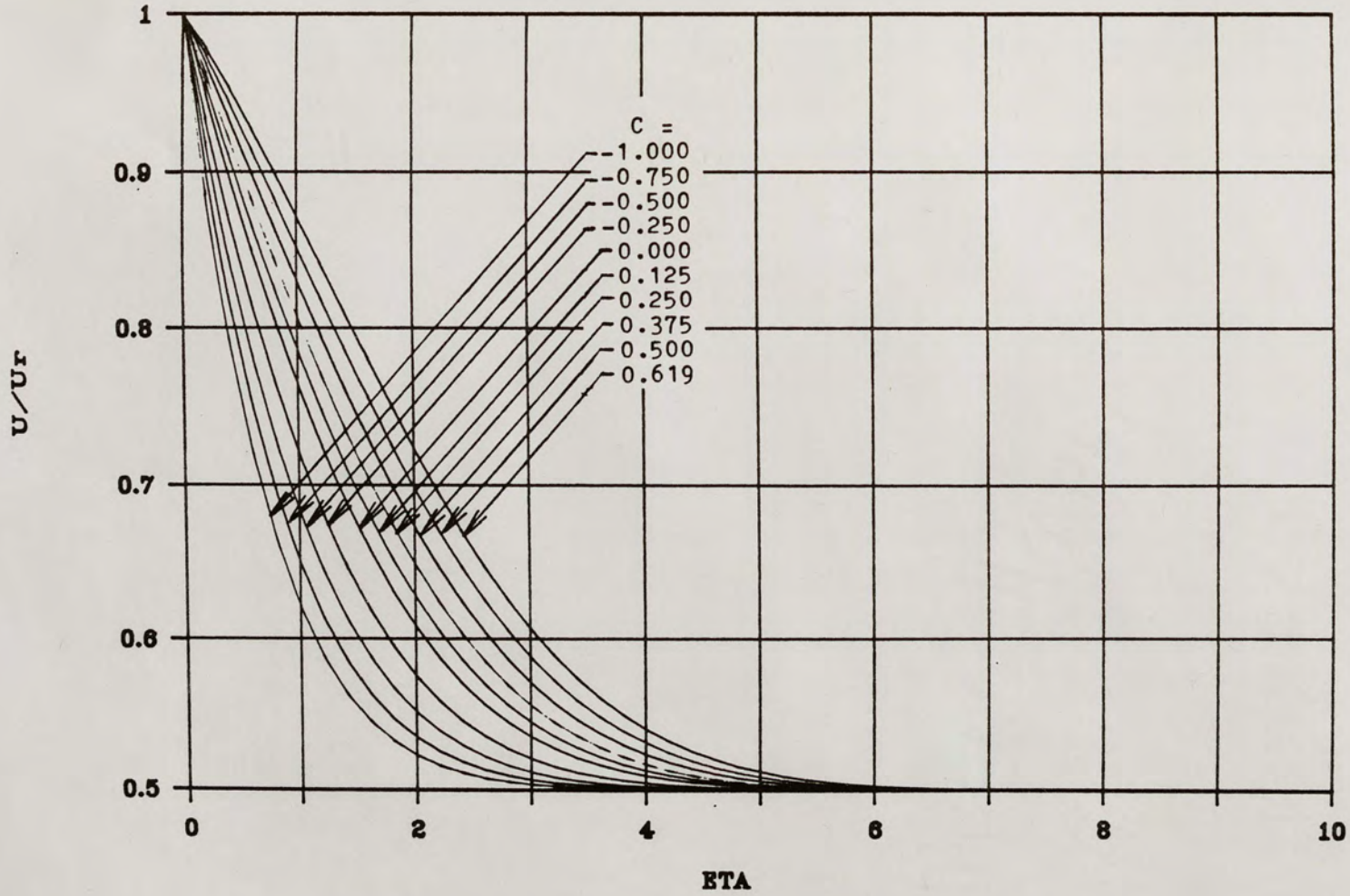

Figure 4. Velocity Profiles for $U_w - U_\infty = 0.5$.

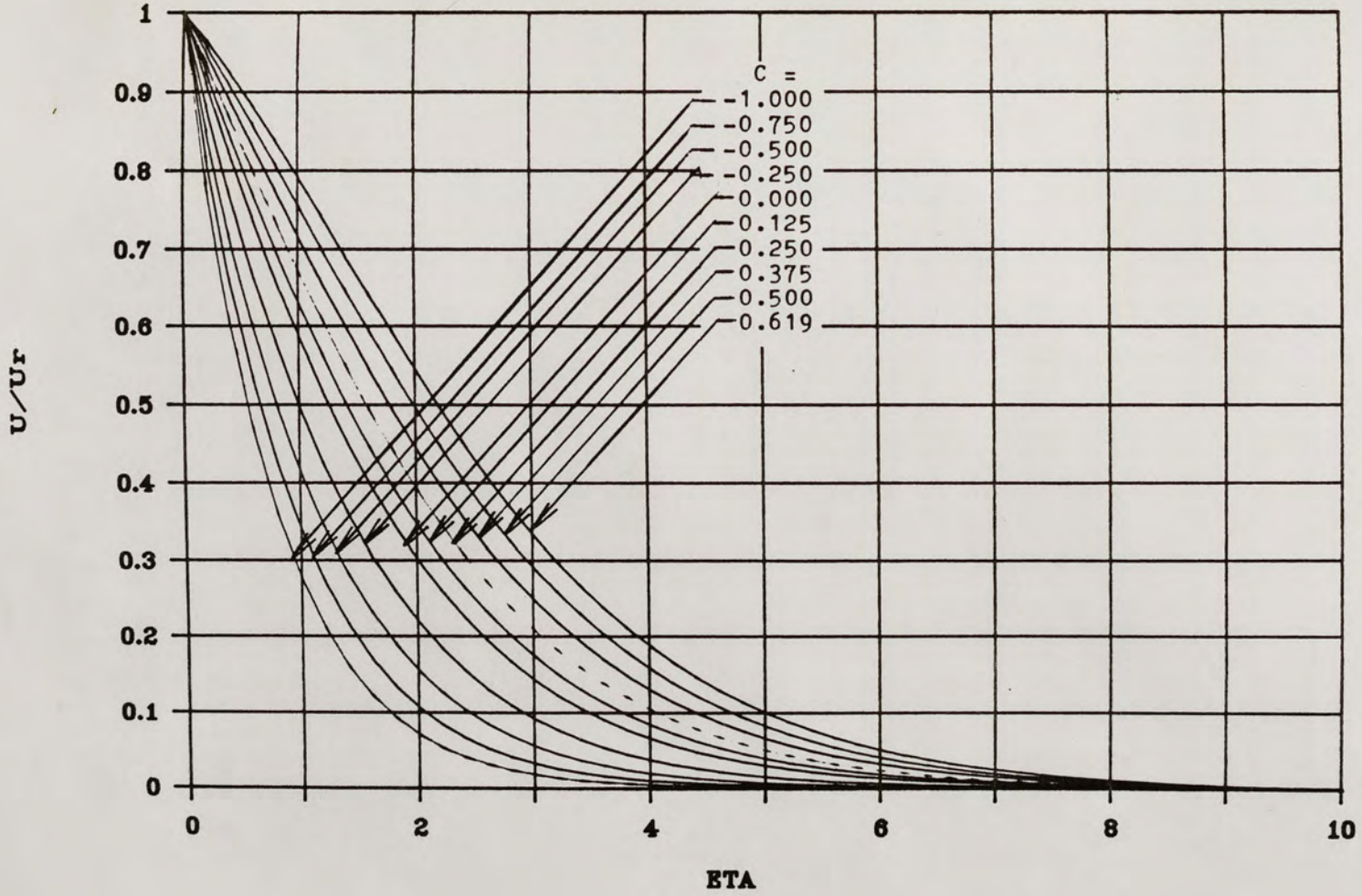

Figure 5. Velocity Profiles for $U_w - U_\infty = 1$.

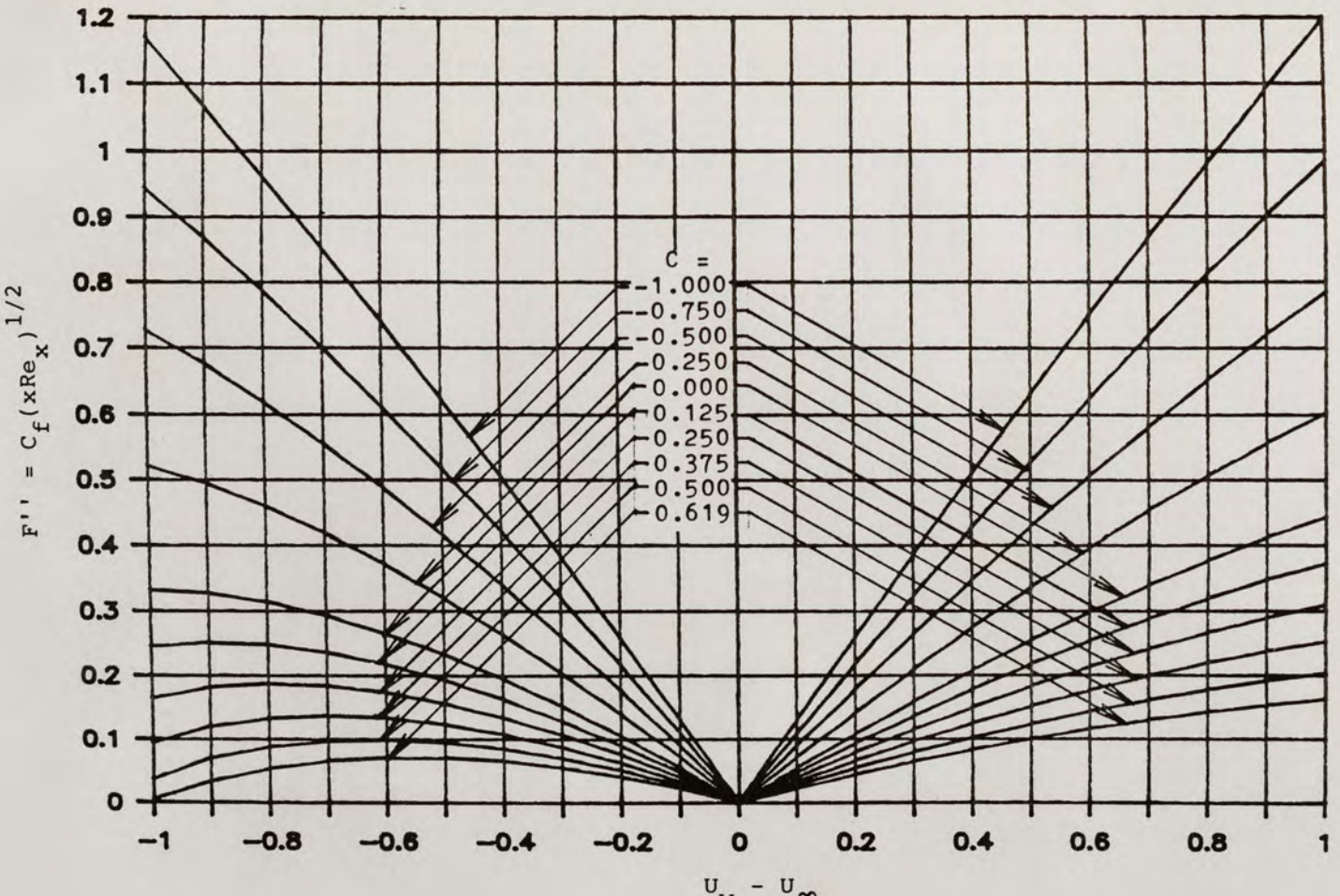

Figure 6. Friction Coefficient vs. Injection Parameter and Velocity Difference.

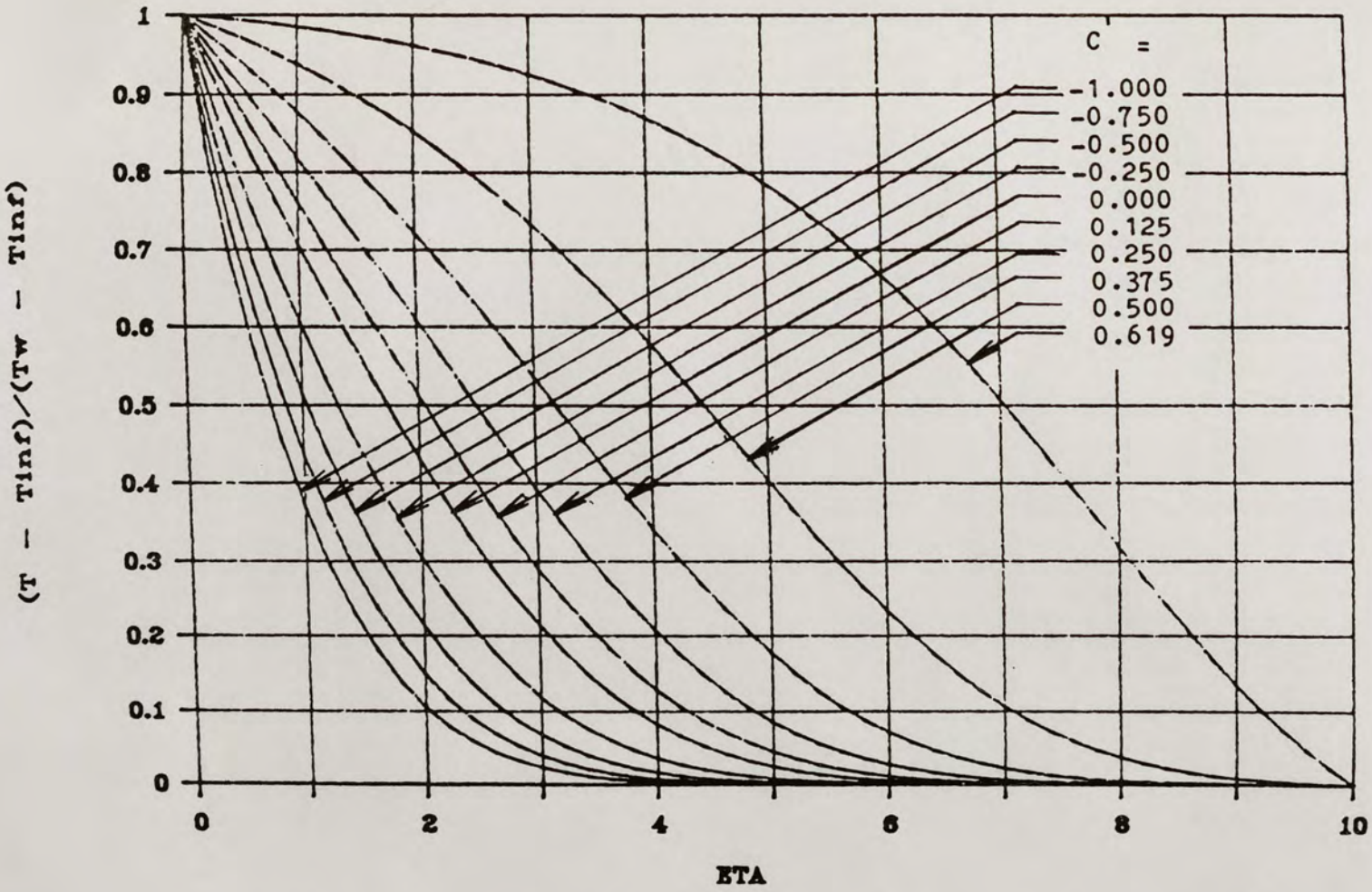

Figure 7. Temperature Profiles for $U_\infty - U_w = 1$ and Pr = 0.72.

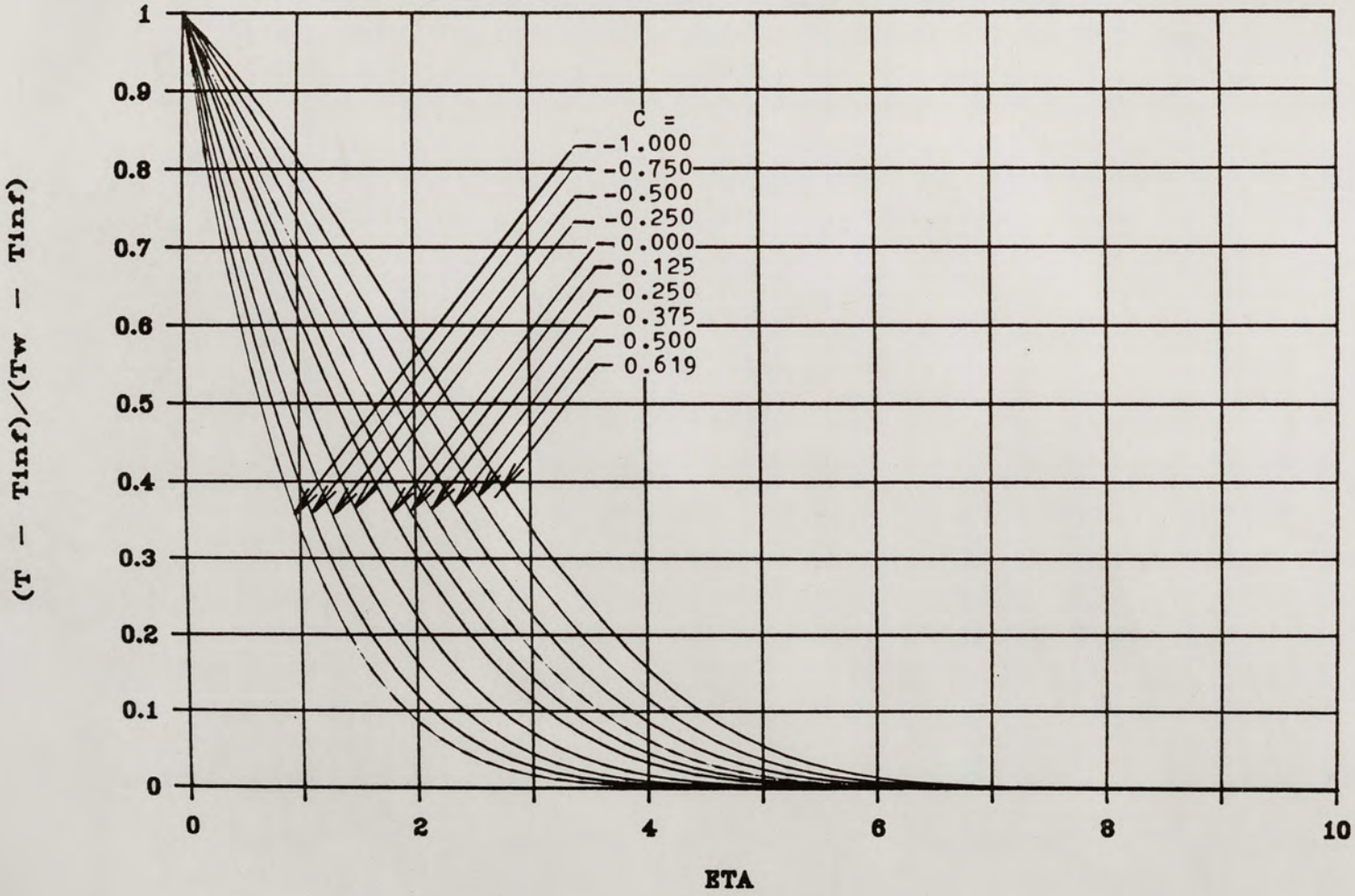

Figure 8. Temperature Profiles for $U_\infty - U_w = 0.5$ and Pr = 0.72.

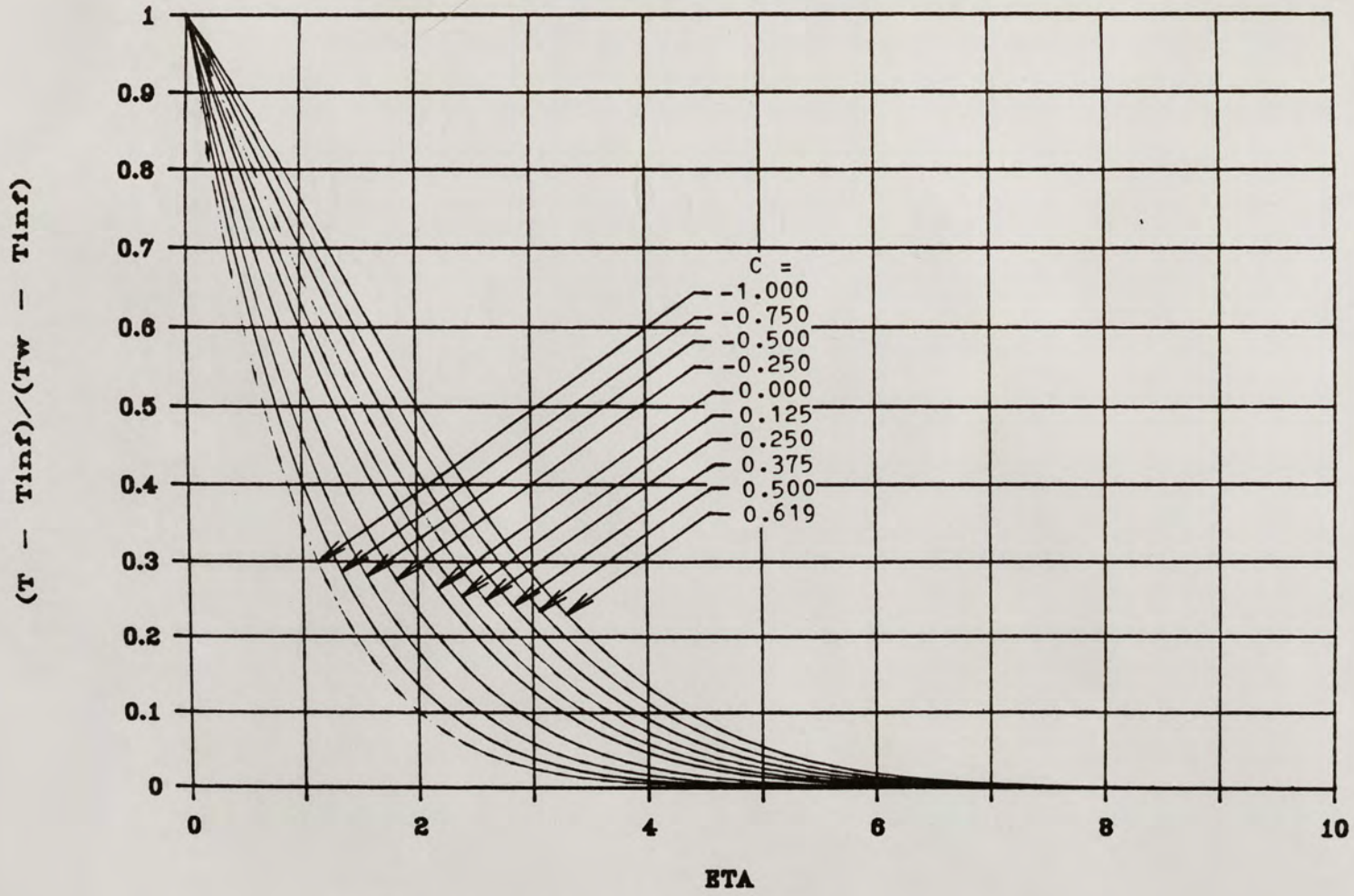

Figure 9. Temperature Profiles for $U_w - U_\infty = 0.5$ and Pr = 0.72.

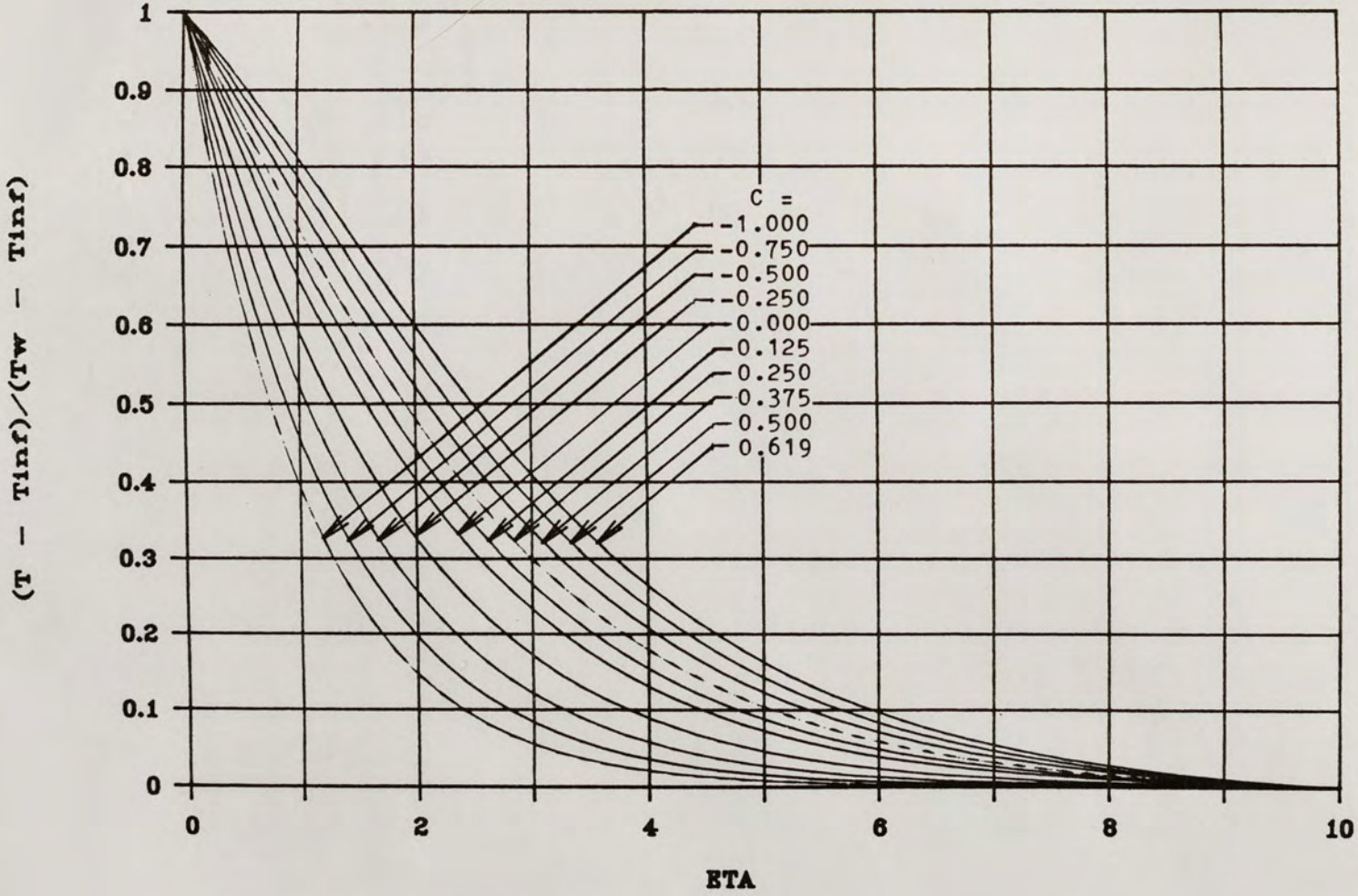

Figure 10. Temperature Profiles for $U_w - U_\infty = 1$ and Pr = 0.72.

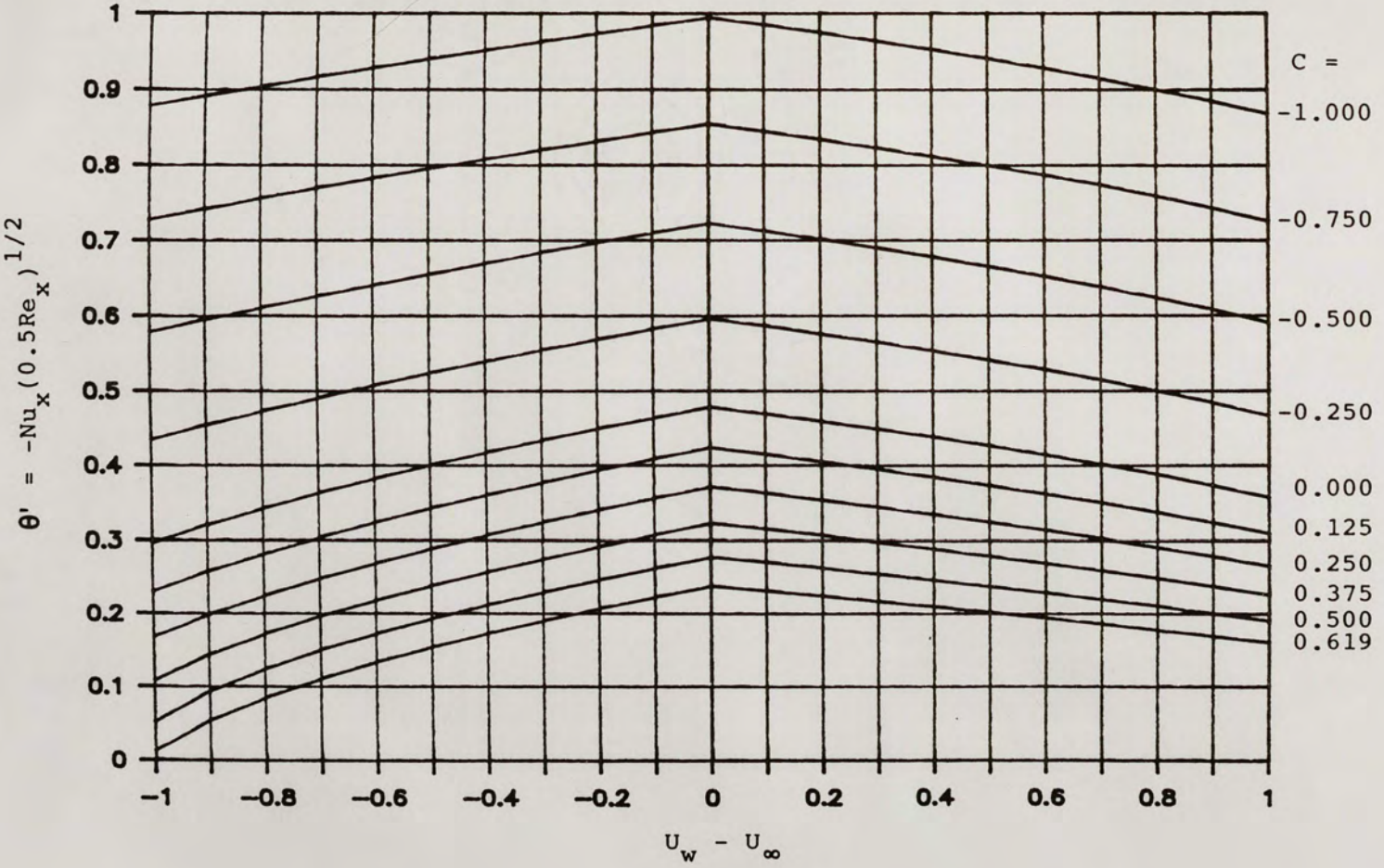

Figure 11. Nusselt Number vs. Injection Parameter and Velocity Difference for Pr = 0.72.

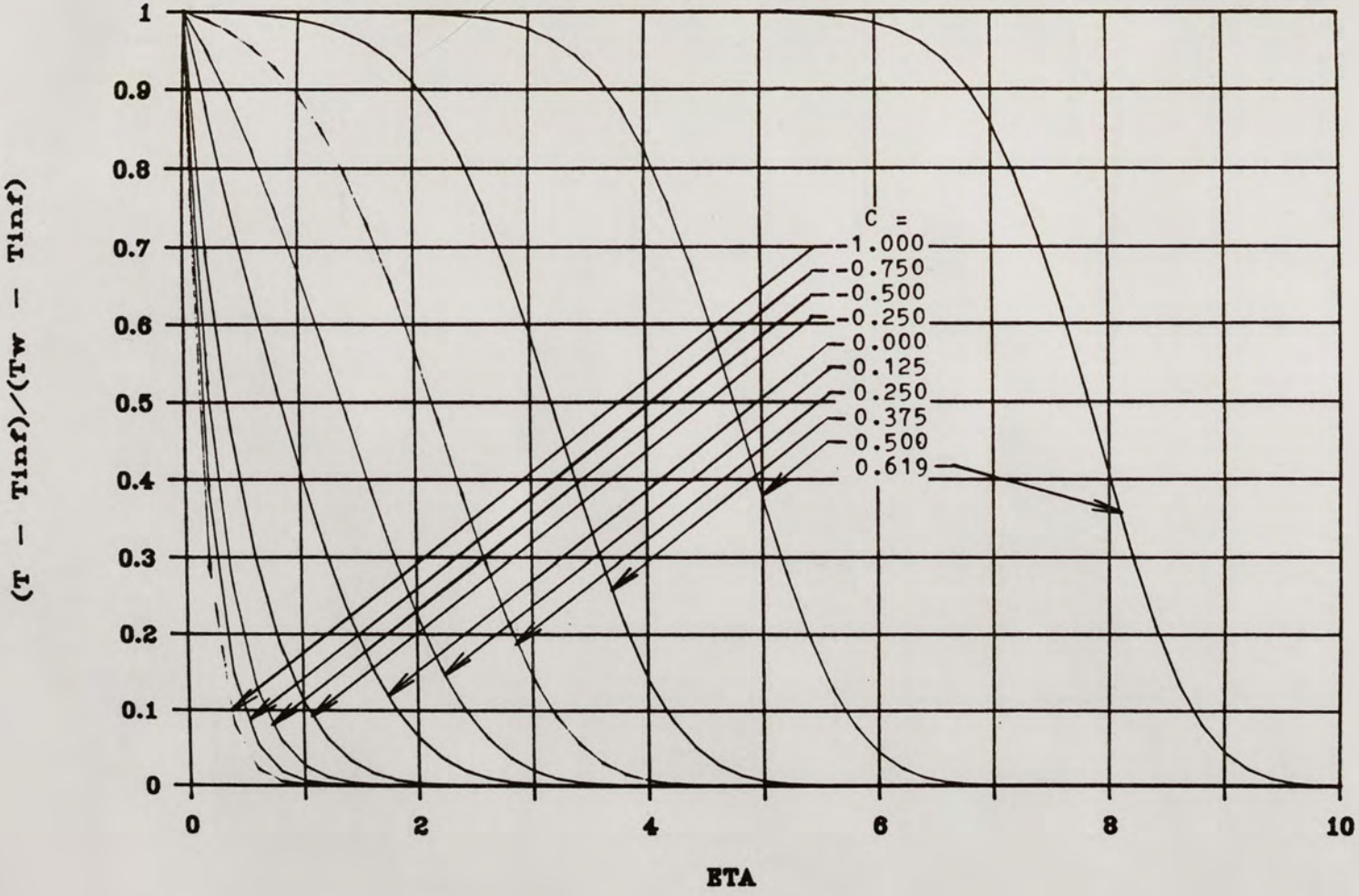

Figure 12. Temperature Profiles for $U_\infty - U_w = 1$ and Pr = 6.0.

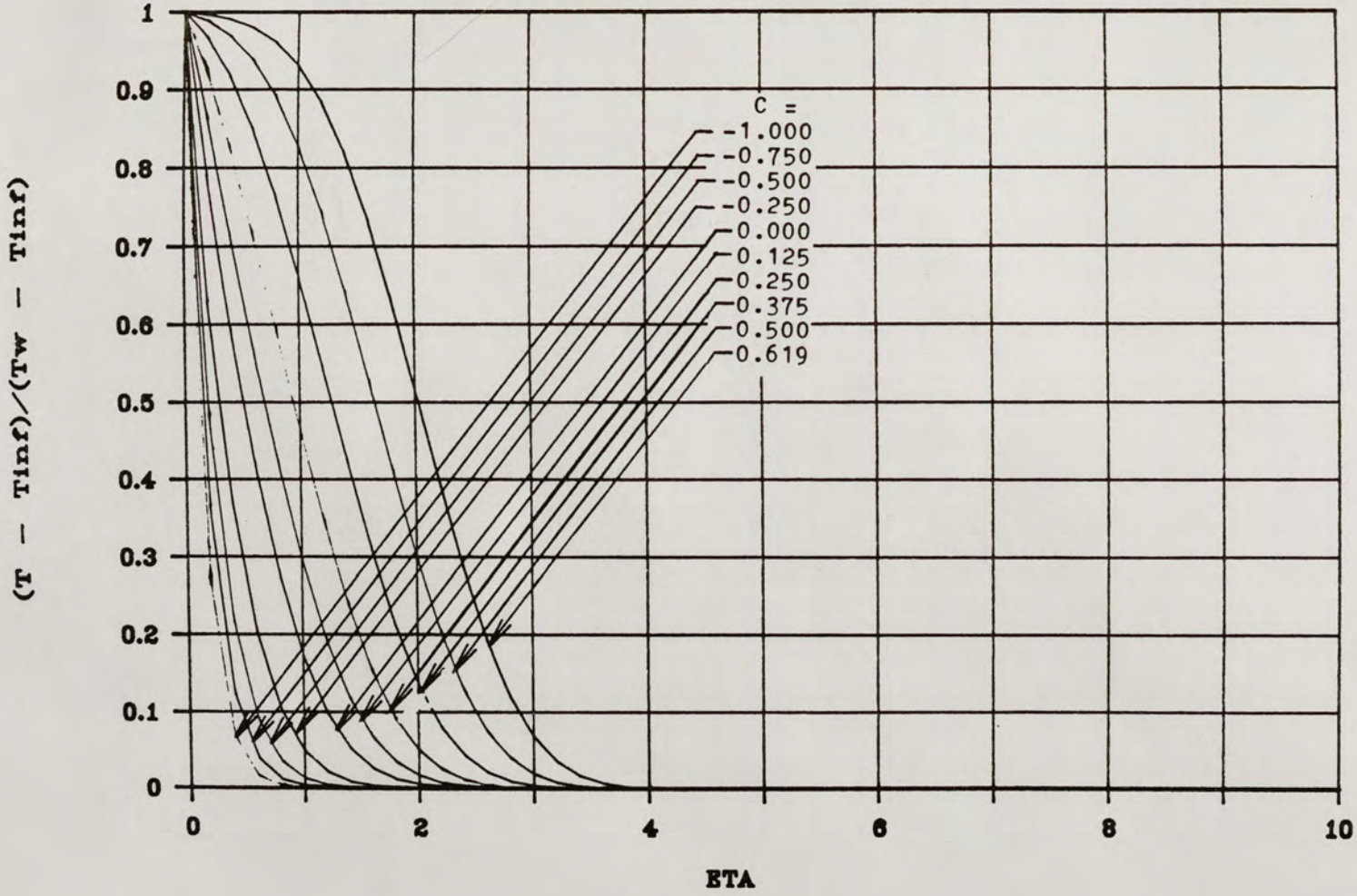

Figure 13. Temperature Profiles for $U_\infty - U_W = 0.5$ and Pr = 6.0.

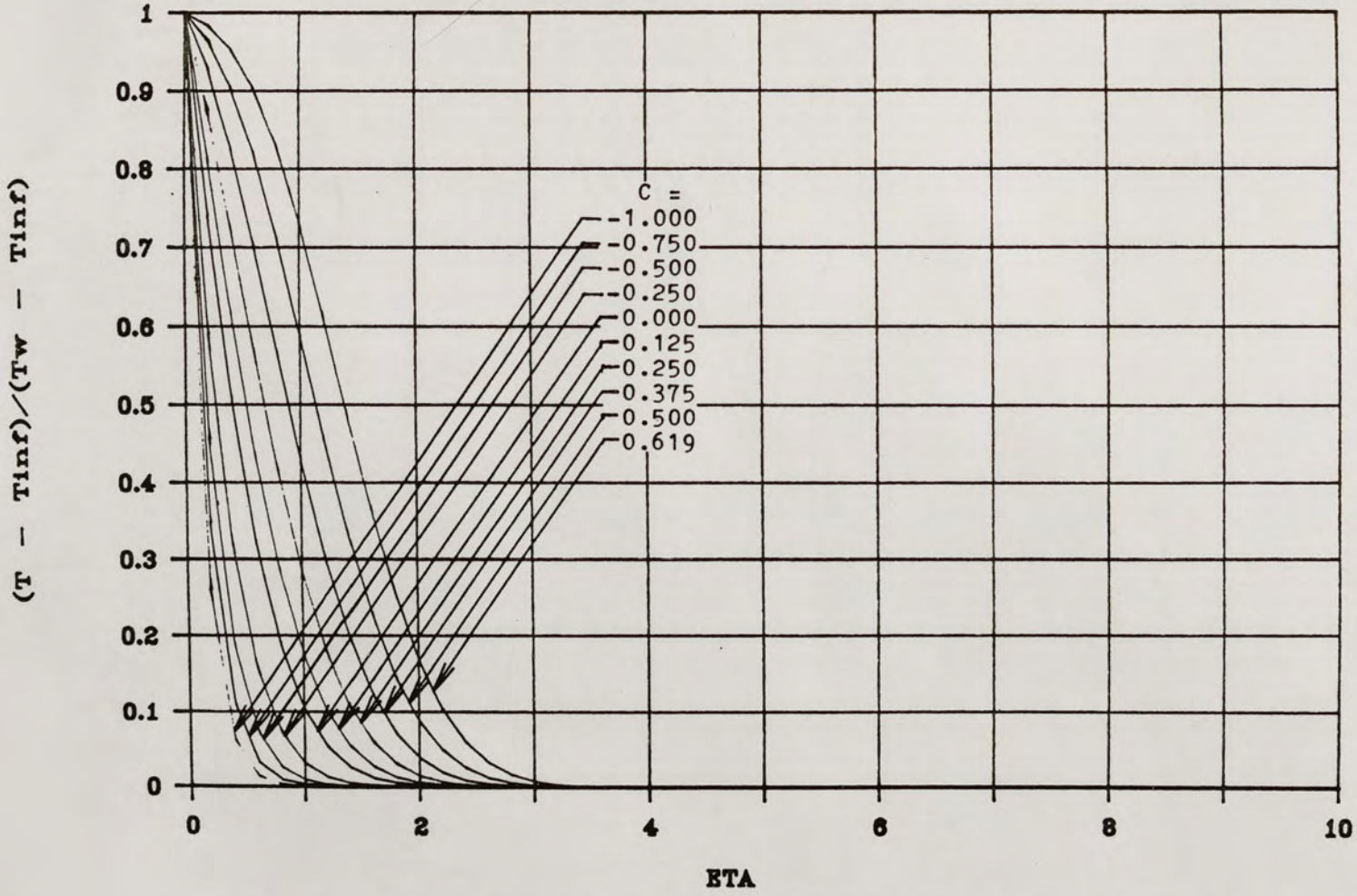

Figure 14. Temperature Profiles for $U_w - U_\infty = 0.5$ and Pr = 6.0.

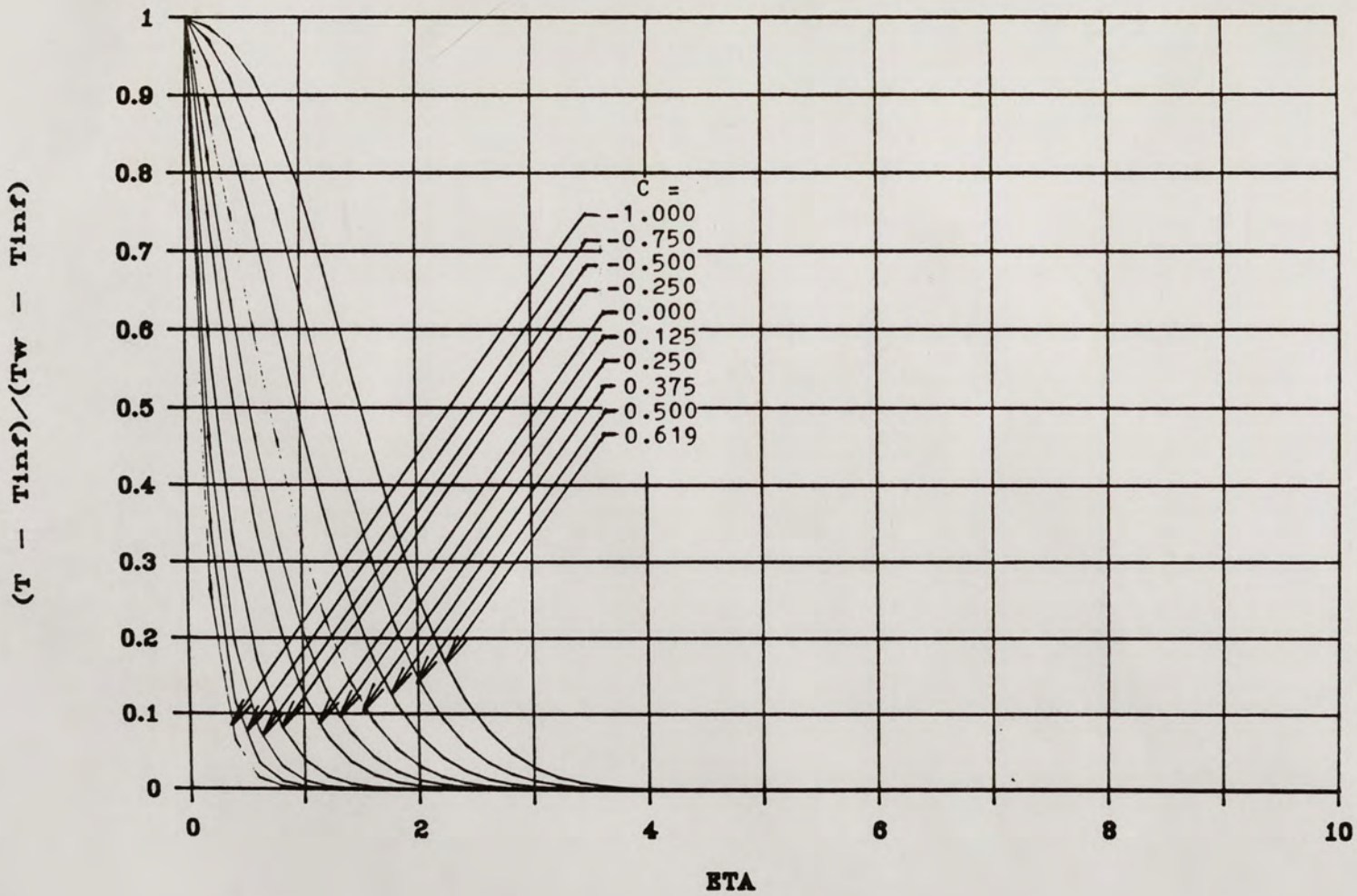

Figure 15. Temperature Profiles for $U_w - U_\infty = 1$ and Pr = 6.0.

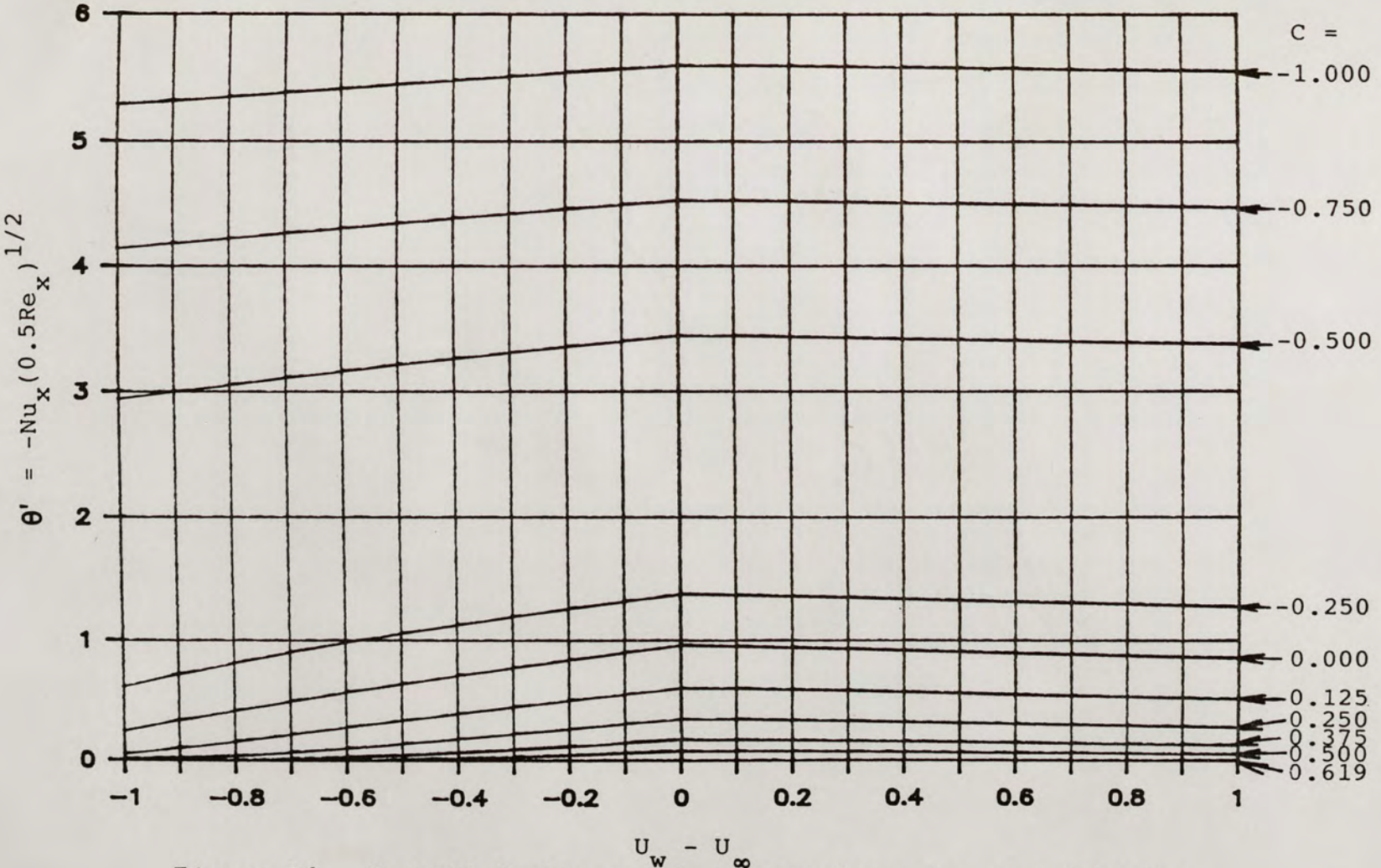

Figure 16. Nusselt Number vs. Injection Parameter and Velocity Differences for Pr = 6.0.

Figures 7 - 10 and 12 - 15 show the thermal boundary layer profiles for cases where $|U_\infty - U_w| = 1$ and 0.5. As with the momentum boundary layer, the thermal boundary layer is thicker for increasing similar injection parameter (C) and velocity difference. By comparing Figures 7 - 10 with 12 - 15, it is seen, as is the case for the thermal boundary layer in the classical Blasius problem, that the thermal boundary layer thickness increases with the Prandtl number, Pr. Figures 11 and 16 show how the Nusselt Number (Nu) varies with velocity difference ($U_w - U_\infty$), injection parameter (C), and Prandtl number (Pr). The most obvious thing that is seen when comparing Figures 11 and 16 is that the Nusselt number varies from about 3 to 6 times greater for Pr = 6 than for Pr = 0.72 for comparable cases. The next thing that is noticed is that the Nusselt number (Nu) increases with decreasing injection parameter (C), velocity difference $|U_w - U_\infty|$, and increasing Prandtl number (Pr). The increased Nu with increasing Pr is seen in equation (6). Tables 1 - 3 give the data from Figures 6, 11 and 16. Comparing data from Tables 1 and 2 with Figures 2 and 3 from Abdelhafez [1] show good agreement.

TABLE 1

FRICTION COEFFICIENT VS. VELOCITY DIFFERENCE AND INJECTION PARAMETER

($0.5C_f(Re_x)^{1/2}$ VS. $U_w - U_\infty$ AND C)

$U_w - U_\infty$	Values of C									
	-1.0000	-0.7500	-0.5000	-0.2500	0.0000	0.1250	0.2500	0.3750	0.5000	0.6190
0.00	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
0.10	0.1309	0.1102	0.0905	0.0722	0.0554	0.0477	0.0406	0.0340	0.0281	0.0231
-0.10	0.1306	0.1098	0.0900	0.0715	0.0546	0.0469	0.0397	0.0330	0.0271	0.0220
0.20	0.2597	0.2182	0.1788	0.1421	0.1086	0.0934	0.0793	0.0663	0.0546	0.0447
-0.20	0.2584	0.2166	0.1767	0.1394	0.1054	0.0898	0.0754	0.0622	0.0503	0.0403
0.30	0.3862	0.3239	0.2647	0.2096	0.1597	0.1370	0.1160	0.0968	0.0795	0.0649
-0.30	0.3834	0.3202	0.2600	0.2036	0.1522	0.1287	0.1070	0.0872	0.0695	0.0547
0.40	0.5104	0.4271	0.3482	0.2748	0.2084	0.1784	0.1506	0.1254	0.1027	0.0836
-0.40	0.5054	0.4206	0.3397	0.2638	0.1946	0.1631	0.1341	0.1077	0.0842	0.0648
0.50	0.6321	0.5278	0.4290	0.3373	0.2548	0.2175	0.1832	0.1521	0.1242	0.1009
-0.50	0.6244	0.5176	0.4155	0.3198	0.2325	0.1928	0.1563	0.1233	0.0942	0.0704
0.60	0.7513	0.6258	0.5070	0.3972	0.2985	0.2542	0.2135	0.1768	0.1440	0.1166
-0.60	0.7402	0.6110	0.4873	0.3712	0.2652	0.2171	0.1730	0.1334	0.0988	0.0710
0.70	0.8677	0.7208	0.5821	0.4541	0.3396	0.2884	0.2416	0.1994	0.1620	0.1308
-0.70	0.8527	0.7007	0.5549	0.4177	0.2924	0.2356	0.1836	0.1372	0.0972	0.0658
0.80	0.9812	0.8128	0.6540	0.5078	0.3777	0.3198	0.2671	0.2198	0.1781	0.1435
-0.80	0.9619	0.7865	0.6179	0.4589	0.3134	0.2474	0.1871	0.1337	0.0884	0.0540
0.90	1.0915	0.9013	0.7222	0.5580	0.4126	0.3483	0.2900	0.2380	0.1922	0.1545
-0.90	1.0675	0.8681	0.6760	0.4942	0.3270	0.2511	0.1817	0.1207	0.0700	0.0339
1.00	1.1982	0.9860	0.7864	0.6041	0.4439	0.3735	0.3100	0.2536	0.2044	0.1639
-1.00	1.1694	0.9454	0.7289	0.5228	0.3321	0.2447	0.1645	0.0936	0.0356	0.0046

TABLE 2

NUSSELT NUMBER VS. INJECTION PARAMETER AND VELOCITY DIFFERENCE FOR AIR

($\theta' = - Nu_x (Re_x)^{-1/2}$ FOR Pr = 0.72)

$U_w - U_\infty$	Values of C									
	-1.0000	-0.7500	-0.5000	-0.2500	0.0000	0.1250	0.2500	0.3750	0.5000	0.6190
0.00	0.9946	0.8558	0.7225	0.5962	0.4787	0.4239	0.3720	0.3233	0.2780	0.2382
0.10	0.9843	0.8452	0.7119	0.5858	0.4688	0.4143	0.3629	0.3148	0.2701	0.2310
-0.10	0.9841	0.8443	0.7100	0.5827	0.4643	0.4091	0.3570	0.3080	0.2627	0.2230
0.20	0.9736	0.8343	0.7009	0.5750	0.4585	0.4045	0.3536	0.3061	0.2621	0.2238
-0.20	0.9733	0.8326	0.6972	0.5688	0.4494	0.3938	0.3412	0.2921	0.2467	0.2071
0.30	0.9626	0.8230	0.6895	0.5638	0.4480	0.3944	0.3441	0.2972	0.2540	0.2164
-0.30	0.9624	0.8206	0.6840	0.5544	0.4338	0.3777	0.3248	0.2754	0.2300	0.1905
0.40	0.9511	0.8112	0.6776	0.5522	0.4370	0.3839	0.3342	0.2881	0.2457	0.2089
-0.40	0.9512	0.8082	0.6704	0.5395	0.4176	0.3609	0.3075	0.2577	0.2121	0.1728
0.50	0.9392	0.7989	0.6652	0.5400	0.4256	0.3731	0.3240	0.2787	0.2372	0.2013
-0.50	0.9398	0.7956	0.6564	0.5239	0.4005	0.3431	0.2891	0.2389	0.1931	0.1539
0.60	0.9267	0.7860	0.6522	0.5273	0.4136	0.3617	0.3135	0.2690	0.2284	0.1935
-0.60	0.9281	0.7826	0.6418	0.5077	0.3825	0.3243	0.2695	0.2187	0.1726	0.1335
0.70	0.9136	0.7724	0.6384	0.5138	0.4010	0.3498	0.3024	0.2588	0.2193	0.1854
-0.70	0.9162	0.7692	0.6268	0.4907	0.3634	0.3040	0.2482	0.1966	0.1500	0.1109
0.80	0.8997	0.7579	0.6236	0.4993	0.3876	0.3372	0.2907	0.2483	0.2099	0.1772
-0.80	0.9039	0.7554	0.6111	0.4728	0.3428	0.2820	0.2248	0.1720	0.1244	0.0853
0.90	0.8848	0.7422	0.6076	0.4836	0.3732	0.3237	0.2784	0.2372	0.2002	0.1687
-0.90	0.8914	0.7411	0.5947	0.4537	0.3204	0.2577	0.1984	0.1433	0.0939	0.0541
1.00	0.8685	0.7249	0.5897	0.4662	0.3574	0.3092	0.2653	0.2256	0.1901	0.1601
-1.00	0.8785	0.7263	0.5775	0.4333	0.2956	0.2300	0.1670	0.1071	0.0511	0.0119

TABLE 3

NUSSELT NUMBER VS. INJECTION PARAMETER AND VELOCITY DIFFERENCE FOR WATER

($\theta' = - Nu_x (Re_x)^{-1/2}$ FOR Pr = 6.0)

$U_w - U_\infty$	Values of C									
	-1.0000	-0.7500	-0.5000	-0.2500	0.0000	0.1250	0.2500	0.3750	0.5000	0.6190
0.00	5.5991	4.5340	3.4448	2.3684	1.3813	0.9598	0.6107	0.3478	0.1731	0.0771
0.10	5.5938	4.5275	3.4366	2.3583	1.3698	0.9483	0.6001	0.3392	0.1672	0.0736
-0.10	5.5700	4.4986	3.4014	2.3160	1.3228	0.9020	0.5582	0.3053	0.1436	0.0596
0.20	5.5885	4.5210	3.4285	2.3483	1.3583	0.9369	0.5898	0.3309	0.1614	0.0702
-0.20	5.5404	4.4625	3.3567	2.2616	1.2617	0.8420	0.5043	0.2629	0.1156	0.0440
0.30	5.5833	4.5146	3.4204	2.3383	1.3469	0.9256	0.5795	0.3226	0.1558	0.0670
-0.30	5.5104	4.4255	3.3107	2.2050	1.1977	0.7794	0.4491	0.2209	0.0893	0.0306
0.40	5.5782	4.5082	3.4125	2.3284	1.3357	0.9144	0.5694	0.3146	0.1504	0.0639
-0.40	5.4799	4.3877	3.2632	2.1459	1.1305	0.7140	0.3924	0.1796	0.0654	0.0195
0.50	5.5731	4.5019	3.4046	2.3187	1.3246	0.9036	0.5596	0.3069	0.1453	0.0610
-0.50	5.4489	4.3490	3.2141	2.0840	1.0595	0.6454	0.3345	0.1396	0.0444	0.0111
0.60	5.5681	4.4957	3.3969	2.3092	1.3138	0.8929	0.5501	0.2994	0.1403	0.0583
-0.60	5.4172	4.3093	3.1631	2.0188	0.9840	0.5731	0.2754	0.1018	0.0269	0.0052
0.70	5.5631	4.4896	3.3893	2.2999	1.3033	0.8826	0.5409	0.2922	0.1357	0.0557
-0.70	5.3850	4.2685	3.1101	1.9498	0.9031	0.4966	0.2158	0.0674	0.0137	0.0018
0.80	5.5582	4.4836	3.3819	2.2909	1.2932	0.8727	0.5322	0.2855	0.1313	0.0534
-0.80	5.3521	4.2265	3.0548	1.8762	0.8157	0.4154	0.1565	0.0380	0.0050	0.0004
0.90	5.5534	4.4778	3.3747	2.2822	1.2836	0.8634	0.5240	0.2793	0.1274	0.0513
-0.90	5.3186	4.1832	2.9968	1.7971	0.7199	0.3288	0.0996	0.0158	0.0009	0.0000
1.00	5.5487	4.4721	3.3678	2.2740	1.2747	0.8549	0.5166	0.2736	0.1238	0.0494
-1.00	5.2843	4.1385	2.9357	1.7111	0.6133	0.2366	0.0487	0.0031	0.0000	0.0000

Non-Similar Boundary Layer Case

The second part of this thesis describes the calculation of the momentum and thermal boundary layers for the moving flat plate with uniform mass transfer at the plate surface. This boundary condition is an example of a non-similar boundary condition. It is called non-similar because the boundary condition and governing equations cannot both be expressed in terms of a single variable or parameter.

Equations (1) - (3) are solved and repeated here for convenience.

$$u_x + v_y = 0 \tag{1}$$

$$uu_x + vu_y = \nu u_{yy} \tag{2}$$

and

$$uT_x + vT_y = (T_{yy})/Pr. \tag{3}$$

The boundary conditions for this flow field are given as

$$@\ y = 0, \eta = 0;\ u = U_w,\ v = v_w,\ T = T_w$$

$$@\ y = \infty, \eta = \infty;\ u = U_\infty$$

where in this case v_w = constant.

The solution is obtained by assuming a stream function as follows:

$$\Psi = (2\nu U_r x)^{1/2} f(\zeta, \eta)$$

where

$$U_r = U_\infty \text{ if } U_\infty > U_w$$

$$U_r = U_w \text{ if } U_w > U_\infty$$

(as above), and

$$\zeta = (v_w/U_r)(2U_r x/\nu)^{1/2}$$

$$\zeta = v_w/U_r(2Re)^{1/2}$$

$$\eta = y(U_r/2\nu x)^{1/2}.$$

It is known that the stream function, Ψ, satisfies the continuity equation. It is also known that $u = \Psi_y$ and $v = -\Psi_x$.

Performing the necessary differentiations to obtain u, u_x, u_y, u_{yy} and v yields:

$$u = U_r f' \tag{11}$$

$$u_x = (U_r/2x)(\zeta f' - \eta f'') \tag{12}$$

$$u_y = (U_r/2x)(2U_r x/\nu)^{1/2} f'' \tag{13}$$

$$u_{yy} = (U_r/2x)(U_r/\nu) f''' \tag{14}$$

$$v = -.5(2\nu U_r/x)^{1/2}(f + \zeta f_\zeta - \eta f'). \tag{15}$$

Substituting equations (11) - (15) into equation (2) yields:

$$f''' + ff'' = \zeta(f'f'_\zeta - f''f_\zeta). \tag{16}$$

Note: Primes denote differentiation with respect to η.

From equation (11) and the boundary conditions, it follows that

$$f'(\zeta,0) = U_w/U_r. \tag{17}$$

Substituting $\eta = 0$, into equation (15), we get

$$f_\zeta + (1/\zeta)f = -1$$

which can be solved by use of an integrating factor to obtain

$$f(\zeta,0) = -\zeta/2.$$

Finally, at $\eta = \infty$, $u = U_\infty$, taken with equation (11), the final boundary condition is found to be

$$f'(\zeta,\infty) = U_\infty/U_r. \tag{19}$$

If we let the partial derivative of f with respect to ζ be represented by g, rewriting equation (16), we get

$$f''' + ff'' = \zeta(f'g' - f''g). \tag{20}$$

Next, the local non-similarity method described in Sparrow, Quack and Boerner [7] is applied. This is achieved by differentiating equation (20) with respect to ζ. This result is

$$g''' + fg'' - f'g' + 2f''g = 0. \tag{21}$$

The boundary conditions for this equation are found to be

$$g(\zeta,0) = -1/2;\ g'(\zeta,0) = 0;\ g'(\zeta,\infty) = 0.$$

To solve the energy equation, a non-dimensional variable, Theta, θ, is defined as

$$\theta(\zeta,\eta) = (T - T_\infty)/(T_w - T_\infty). \tag{22}$$

This implies that $T = (T_w - T_\infty)\theta + T_\infty$, which yields:

$$T_x = (T_w - T_\infty)(1/2x)(\zeta\theta_\zeta - \eta\theta') \tag{23}$$

$$T_y = (T_w - T_\infty)\theta'(U_r/2\nu x)^{1/2} \tag{24}$$

$$T_{yy} = (T_w - T_\infty)\theta''(U_r/2\nu x). \tag{25}$$

Substituting equations (11) - (14) and (23) - (25) into equation (3) results in

$$(1/Pr)\theta'' + f\theta' = \zeta(f'\theta_{\zeta} - \theta' f)$$

or

$$(1/Pr)\theta'' + f\theta' = \zeta(f'\Phi - \theta' g) \qquad (26)$$

where Φ is defined as

$$\Phi = \theta_{\zeta}.$$

The boundary conditions become

$$@\ y = 0,\ \eta = 0;\ \theta(\zeta,0) = 1$$

$$@\ y = \infty,\ \eta = \infty;\ \theta(\zeta,\infty) = 0.$$

For the local non-similarity model differentiate equation (26) with respect to ζ to obtain

$$(1/Pr)\Phi'' + 2g\theta' + \Phi' f - f'\Phi = 0 \qquad (27)$$

where

$$@\ y = 0,\ \eta = 0;\ \Phi(\zeta,0) = 0$$

$$@\ y = \infty,\ \eta = \infty;\ \Phi(\zeta,\infty) = 0.$$

Equations (20), (21), (26) and (27) are then programmed for solution by means of the fourth order Runge-Kutta method. To solve equations (20) and (21) by the Runge-Kutta method, the following values were defined:

$$f' = p \qquad (28a)$$

$$f'' = q \qquad (28b)$$

$$f''' = \zeta(f'g' - f''g) - ff' \qquad (28c)$$

$$g' = r \qquad (28d)$$

$$g'' = s \qquad (28e)$$

$$g''' = f'g' - fg'' - 2f''g \qquad (28f)$$

Equations (28 a-f) are then used to generate the finite difference equations used in the fourth order Runge-Kutta method as follows:

$$\Delta_1 f_i = hp_i \tag{29a}$$

$$\Delta_1 p_i = hq_i \tag{29b}$$

$$\Delta_1 q_i = \zeta(p_i r_i - q_i g_i) - f_i p_i \tag{29c}$$

$$\Delta_1 g_i = hr_i \tag{29d}$$

$$\Delta_1 r_i = hs_i \tag{29e}$$

$$\Delta_1 s_i = p_i r_i - f_i s_i - 2q_i g_i \tag{29f}$$

$$\Delta_2 f_i = h(f_i + 0.5\Delta_1 p_i) \tag{29g}$$

$$\Delta_2 p_i = h(p_i + 0.5\Delta_1 q_i) \tag{29h}$$

$$\begin{aligned}\Delta_2 q_i = h\{\zeta[&(p_i + 0.5\Delta_1 p_i)(r_i + 0.5\Delta_1 r_i) \\ &- (q_i + 0.5\Delta_1 q_i)(g_i + 0.5\Delta_1 g_i)] \\ &- (f_i + 0.5\Delta_1 f_i)(p_i + 0.5\Delta_1 p_i)\}\end{aligned} \tag{29i}$$

$$\Delta_2 g_i = h(r_i + 0.5\Delta_1 r_i) \tag{29j}$$

$$\Delta_2 r_i = h(s_i + 0.5\Delta_1 s_i) \tag{29k}$$

$$\begin{aligned}\Delta_2 s_i = h[&(p_i + 0.5\Delta_1 p_i)(r_i + 0.5\Delta_1 r_i) \\ &- (f_i + 0.5\Delta_1 f_i)(s_i + 0.5\Delta_1 s_i) \\ &- 2(q_i + 0.5\Delta_1 q_i)(g_i + 0.5\Delta_1 g_i)]\end{aligned} \tag{29l}$$

$$\Delta_3 f_i = h(p_i + 0.5\Delta_2 p_i) \tag{29m}$$

$$\Delta_3 p_i = h(q_i + 0.5\Delta_2 q_i) \tag{29n}$$

$$\begin{aligned}\Delta_3 q_i = h\{\zeta[&(p_i + 0.5\Delta_2 p_i)(r_i + 0.5\Delta_2 r_i) \\ &- (q_i + 0.5\Delta_2 q_i)(g_i + 0.5\Delta_2 g_i)] \\ &- (f_i + 0.5\Delta_2 f_i)(p_i + 0.5\Delta_2 p_i)\}\end{aligned} \tag{29o}$$

$$\Delta_3 g_i = h(r_i + 0.5\Delta_2 r_i) \tag{29p}$$

$$\Delta_3 r_i = h(s_i + 0.5\Delta_2 s_i) \tag{29q}$$

$$\Delta_3 s_i = h[(p_i + 0.5\Delta_2 p_i)(r_i + 0.5\Delta_2 r_i) - (f_i + 0.5\Delta_2 f_i)(s_i + 0.5\Delta_2 s_i) - 2(q_i + 0.5\Delta_2 q_i)(g_i + 0.5\Delta_2 g_i)] \tag{29r}$$

$$\Delta_4 f_i = h(p_i + \Delta_3 p_i) \tag{29s}$$

$$\Delta_4 p_i = h(q_i + \Delta_3 q_i) \tag{29t}$$

$$\Delta_4 q_i = h\{\zeta[(p_i + \Delta_3 p_i)(r_i + \Delta_3 r_i) - (q_i + \Delta_3 q_i)(g_i + \Delta_3 g_i)] - (f_i + \Delta_3 f_i)(p_i + \Delta_3 p_i)\} \tag{29u}$$

$$\Delta_4 g_i = h(r_i + \Delta_3 r_i) \tag{29v}$$

$$\Delta_4 r_i = h(s_i + \Delta_4 s_i) \tag{29w}$$

$$\Delta_4 s_i = h[(p_i + \Delta_3 p_i)(r_i + \Delta_3 r_i) - (f_i + \Delta_3 f_i)(s_i + \Delta_3 s_i) - 2(q_i + \Delta_3 q_i)(g_i + \Delta_3 g_i)]. \tag{29x}$$

Using the values computed in equations (29 a-x), values for f, f', f'', g, g' and g'' are computed at i+1 by the following relations:

$$f_{i+1} = f_i + (\Delta_1 f_i + 2(\Delta_2 f_i + \Delta_3 f_i) + \Delta_4 f_i)/6.0 \tag{30a}$$

$$p_{i+1} = p_i + (\Delta_1 p_i + 2(\Delta_2 p_i + \Delta_3 p_i) + \Delta_4 p_i)/6.0 \tag{30b}$$

$$q_{i+1} = q_i + (\Delta_1 q_i + 2(\Delta_2 q_i + \Delta_3 q_i) + \Delta_4 q_i)/6.0 \tag{30c}$$

$$g_{i+1} = g_i + (\Delta_1 g_i + 2(\Delta_2 g_i + \Delta_3 g_i) + \Delta_4 g_i)/6.0 \tag{30d}$$

$$r_{i+1} = r_i + (\Delta_1 r_i + 2(\Delta_2 r_i + \Delta_3 r_i) + \Delta_4 r_i)/6.0 \tag{30e}$$

$$s_{i+1} = s_i + (\Delta_1 s_i + 2(\Delta_2 s_i + \Delta_3 s_i) + \Delta_4 s_i)/6.0. \tag{30f}$$

Similarly, to solve equations (26) and (27), a set of difference equations are derived as follows:

Let $\theta' = t$ and $\Phi' = u$

$$\Delta_1\theta_i = ht_i \tag{31a}$$

$$\Delta_1 t_i = hPr[\zeta(p_i\Phi_i - t_i g_i) - f_i t_i] \tag{31b}$$

$$\Delta_1\Phi_i = hu_i \tag{31c}$$

$$\Delta_1 u_i = hPr(p_i\Phi_i - f_i u_i - 2g_i t_i) \tag{31d}$$

$$\Delta_2\theta_i = h(t_i + 0.5\Delta_1 t_i) \tag{31e}$$

$$\Delta_2 t_i = hPr\{\zeta[p_i(\Phi_i + 0.5\Delta_1\Phi_i) - g_i(t_i + 0.5\Delta_1 t_i)] - f_i(t_i + 0.5\Delta_1 t_i)\} \tag{31f}$$

$$\Delta_2\Phi_i = h(u_i + 0.5\Delta_1 u_i) \tag{31g}$$

$$\Delta_2 u_i = hPr[p_i(\Phi_i + 0.5\Delta_1\Phi_i) - f_i(u_i + 0.5\Delta_1 u_i) - 2g_i(t_i + 0.5\Delta_1 t_i)] \tag{31h}$$

$$\Delta_3\theta_i = h(t_i + 0.5\Delta_2 t_i) \tag{31i}$$

$$\Delta_3 t_i = hPr\{\zeta[p_i(\Phi_i + 0.5\Delta_2\Phi_i) - g_i(t_i + 0.5\Delta_2 t_i)] - f_i(t_i + 0.5\Delta_2 t_i)\} \tag{31j}$$

$$\Delta_3\Phi_i = h(u_i + 0.5\Delta_2 u_i) \tag{31k}$$

$$\Delta_3 u_i = hPr[p_i(\Phi_i + 0.5\Delta_2\Phi_i) - f_i(u_i + 0.5\Delta_2 u_i) - 2g_i(t_i + 0.5\Delta_2 t_i)] \tag{31l}$$

$$\Delta_4\theta_i = h(t_i + \Delta_3 t_i) \tag{31m}$$

$$\Delta_4 t_i = hPr\{\zeta[p_i(\Phi_i + \Delta_3\Phi_i) - g_i(t_i + \Delta_3 t_i)] - f_i(t_i + \Delta_3 t_i)\}. \tag{31n}$$

$$\Delta_4\Phi_i = h(u_i + \Delta_3 u_i) \tag{31o}$$

$$\Delta_4 u_i = hPr[p_i(\Phi_i + \Delta_3\Phi_i) - f_i(u_i + \Delta_3 u_i) - 2g_i(t_i + \Delta_3 t_i)] \tag{31p}$$

Using equations (31 a-p), θ, θ', Φ, and Φ' at i+1 are obtained by:

$$\theta_{i+1} = \theta_i + (\Delta_1\theta_i + 2(\Delta_2\theta_i + \Delta_3\theta_i) + \Delta_4\theta_i)/6.0 \quad (32a)$$

$$t_{i+1} = t_i + (\Delta_1 t_i + 2(\Delta_2 t_i + \Delta_3 t_i) + \Delta_4 t_i)/6.0 \quad (32b)$$

$$\Phi_{i+1} = \Phi_i + (\Delta_1\Phi_i + 2(\Delta_2\Phi_i + \Delta_3\Phi_i) + \Delta_4\Phi_i)/6.0 \quad (32c)$$

$$u_{i+1} = u_i + (\Delta_1 u_i + 2(\Delta_2 u_i + \Delta_3 u_i) + \Delta_4 u_i)/6.0. \quad (32d)$$

Equations (29 a-x), (30 a-f), (31 a-p) and (32 a-d) are then coded in FORTRAN and input into an IBM XT. Equations (29 a-x) and (30 a-f) are solved first by assuming initial values for f''(0) and g''(0). After computing values for f'(etamax) and g'(etamax), they are compared to the known values of $f'(\infty)$ and $g'(\infty)$, respectively. If those values are within ε of each other, the iteration is stopped and the program continues on to solve equations (31 a-p) and (32 a-d). If f'(etamax) or g'(etamax) are not within ε, then new values of f''(0) or g''(0) are assumed. Again, to make more efficient use of the computer, the secant method is employed to make a new guess for the initial value. If after a large number of iterations (150), convergence did not occur, the case where the solution came closest to converging is used, and computation moves forward to solve equations (31 a-p) and (32 a-d). These equations are solved by the same method except $\theta'(0)$ and $\Phi'(0)$ are assumed. For reasons which I am not able to ascertain, the solution to this set of equations is not achievable. The velocity

profiles for some representative cases are shown in Figures 17 - 20 with the coefficient of friction shown in Figure 21.

As is seen from Figures 17 - 20, the boundary layer thickens with increasing ζ and velocity difference $|U_\infty - U_w|$, as was seen in the similarity case.

Comparing values from Figure 21 at $U_w - U_\infty = -1.0$ with Figure 3 of Sparrow, Quack and Boerner [7] shows good agreement. From Figure 21 it is seen that separation occurs for $U_w - U_\infty = -1.0$ when ζ is approximately 0.4. It is also observed that the friction coefficient is greater when $U_w > U_\infty$ than when $U_w < U_\infty$. Also in Figure 21 it is seen that at $U_w - U_\infty = -0.6$ the curves for $\zeta = -0.2$ and -0.1 meet. This is due to round-off error.

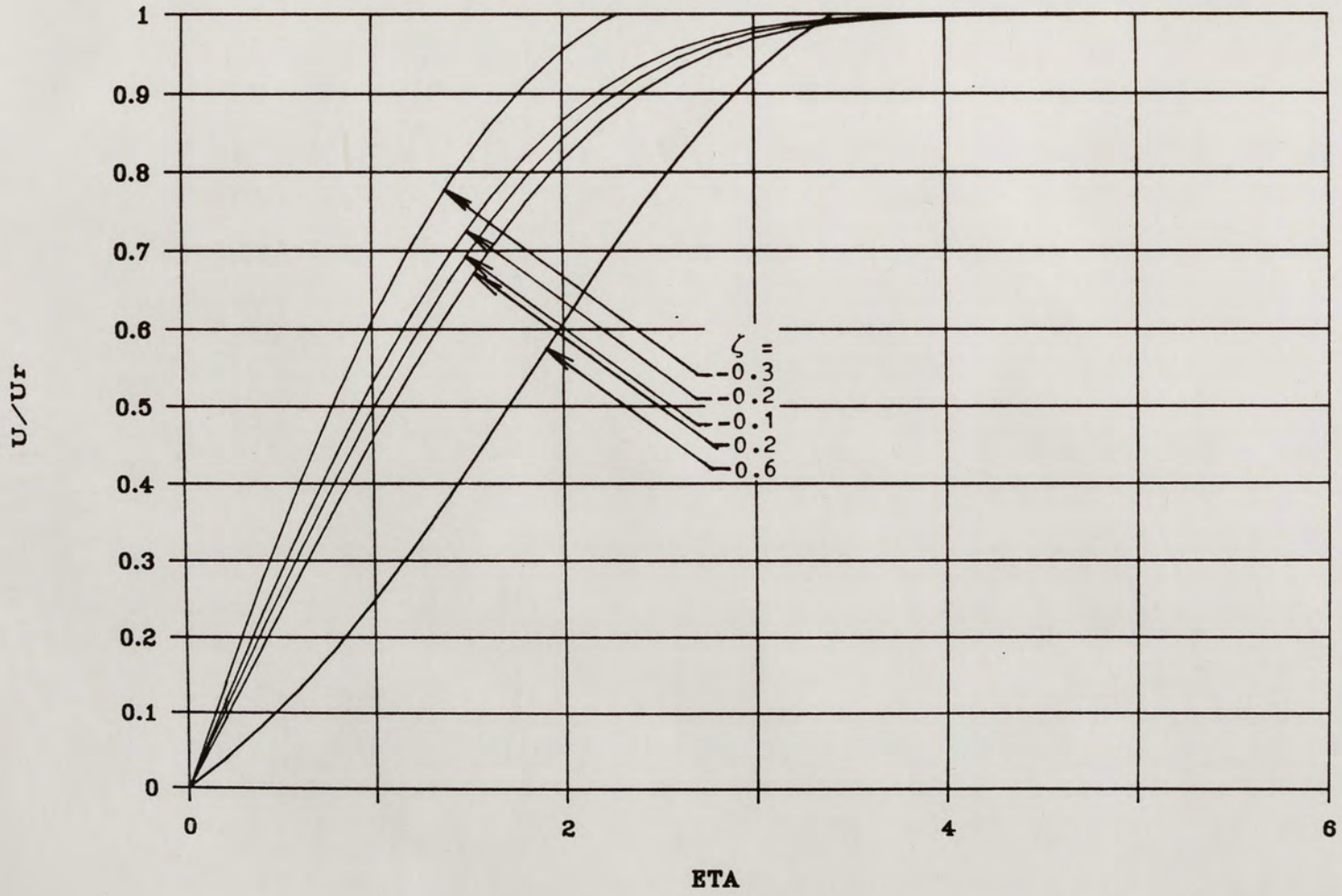

Figure 17. Velocity Profiles for $U_\infty - U_w = 1$ as a Function of ζ.

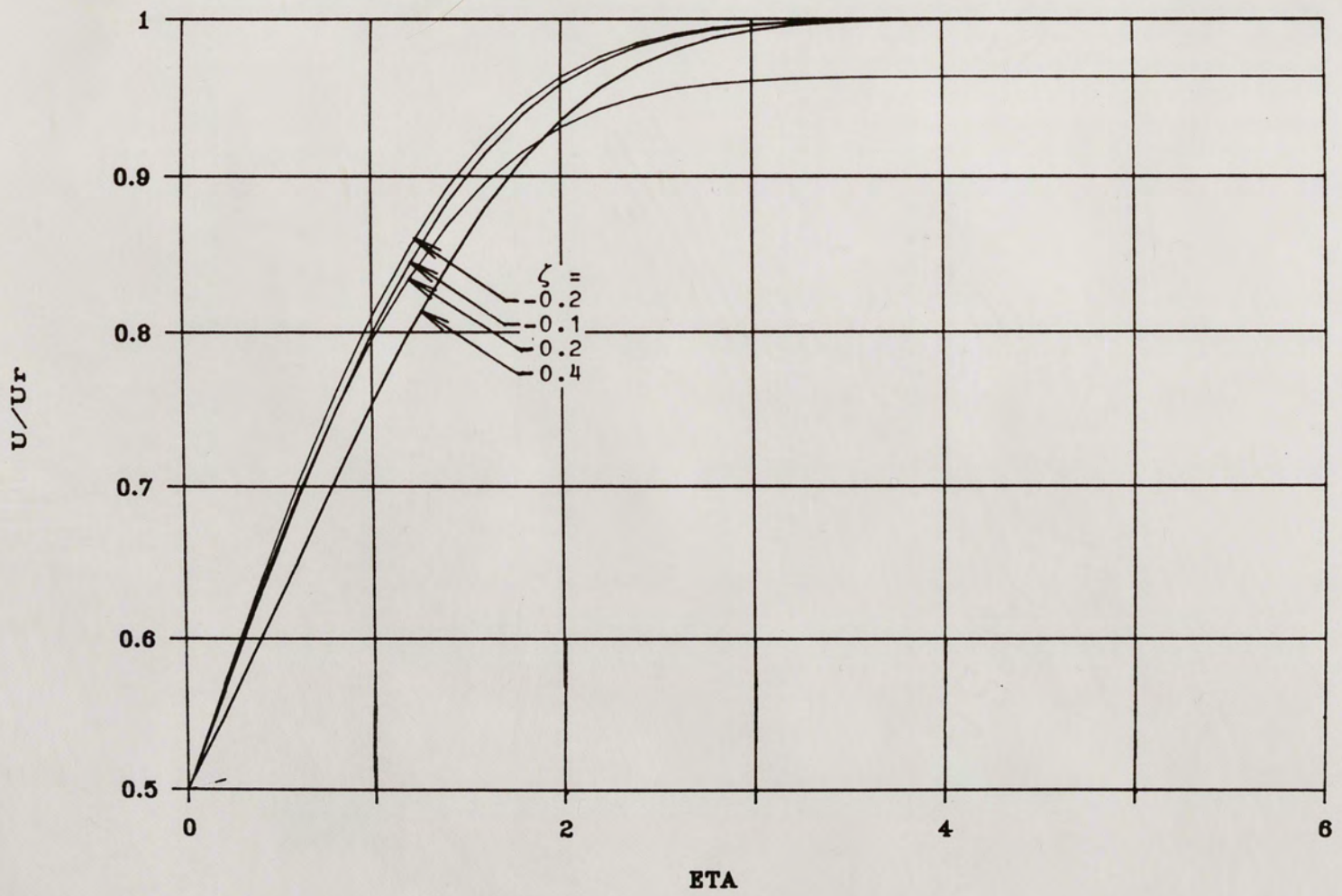

Figure 18. Velocity Profiles for $U_\infty - U_W = 0.5$.

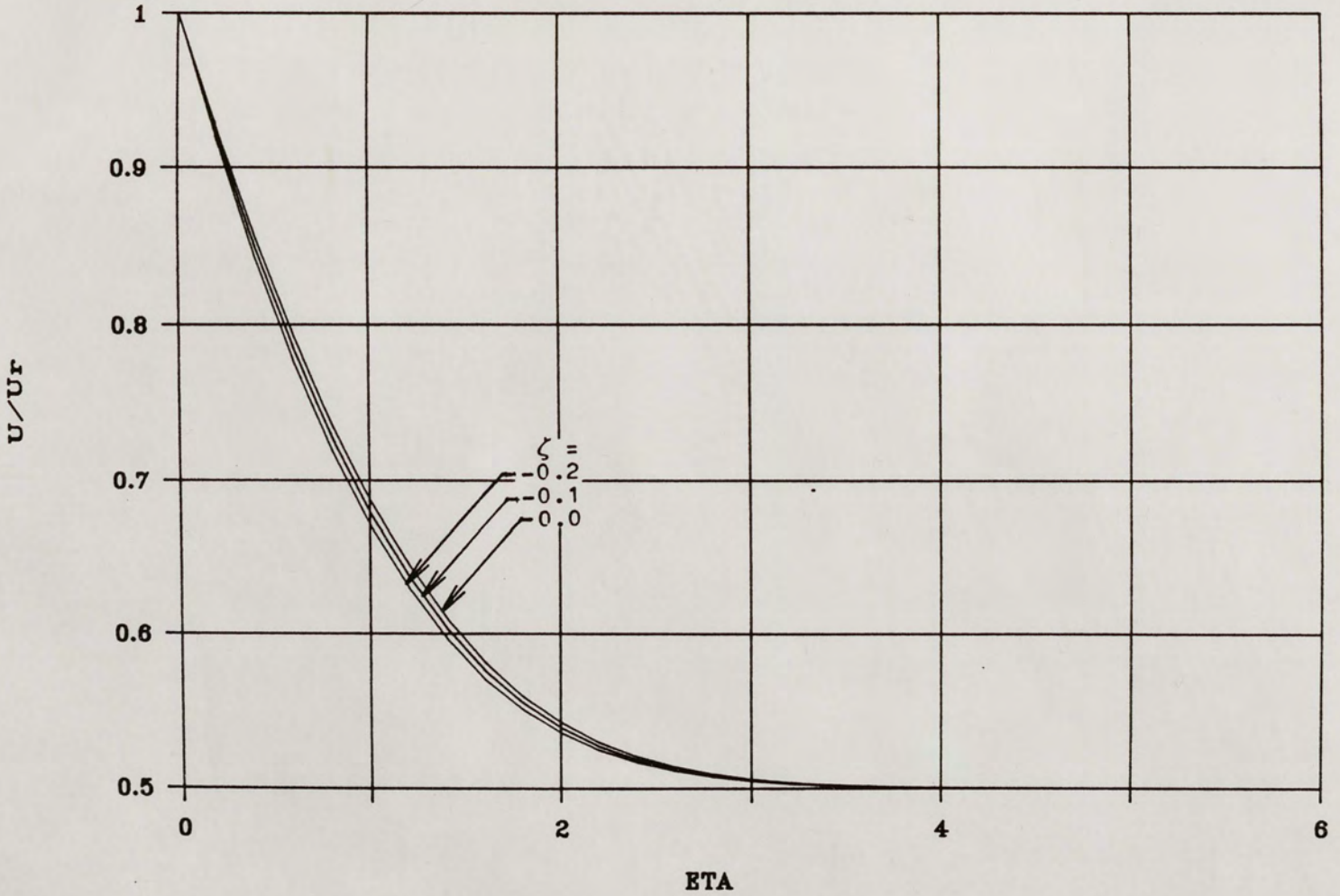

Figure 19. Velocity Profiles for $U_w - U_\infty = 0.5$.

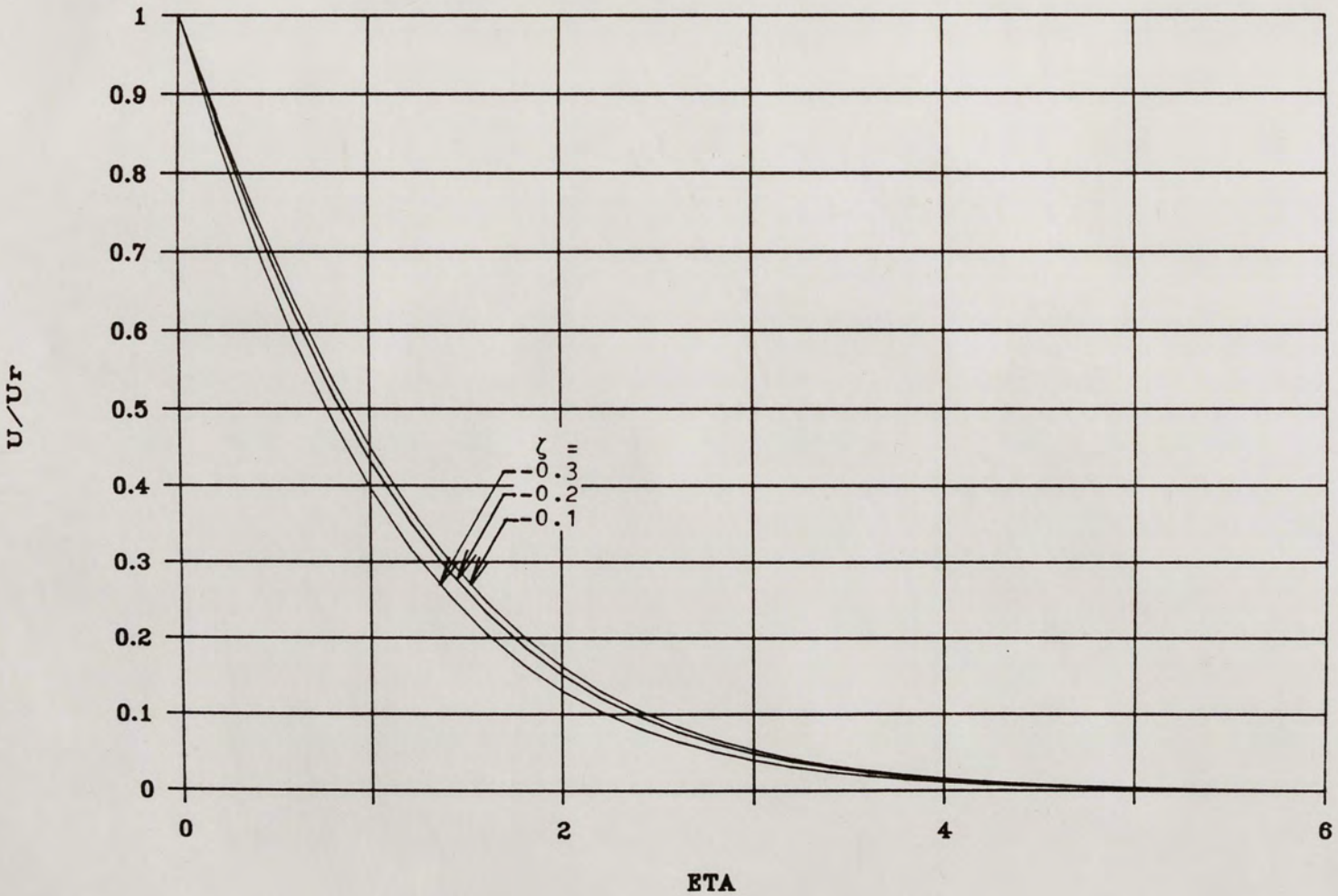

Figure 20. Velocity Profiles for $U_w - U_\infty = 1$.

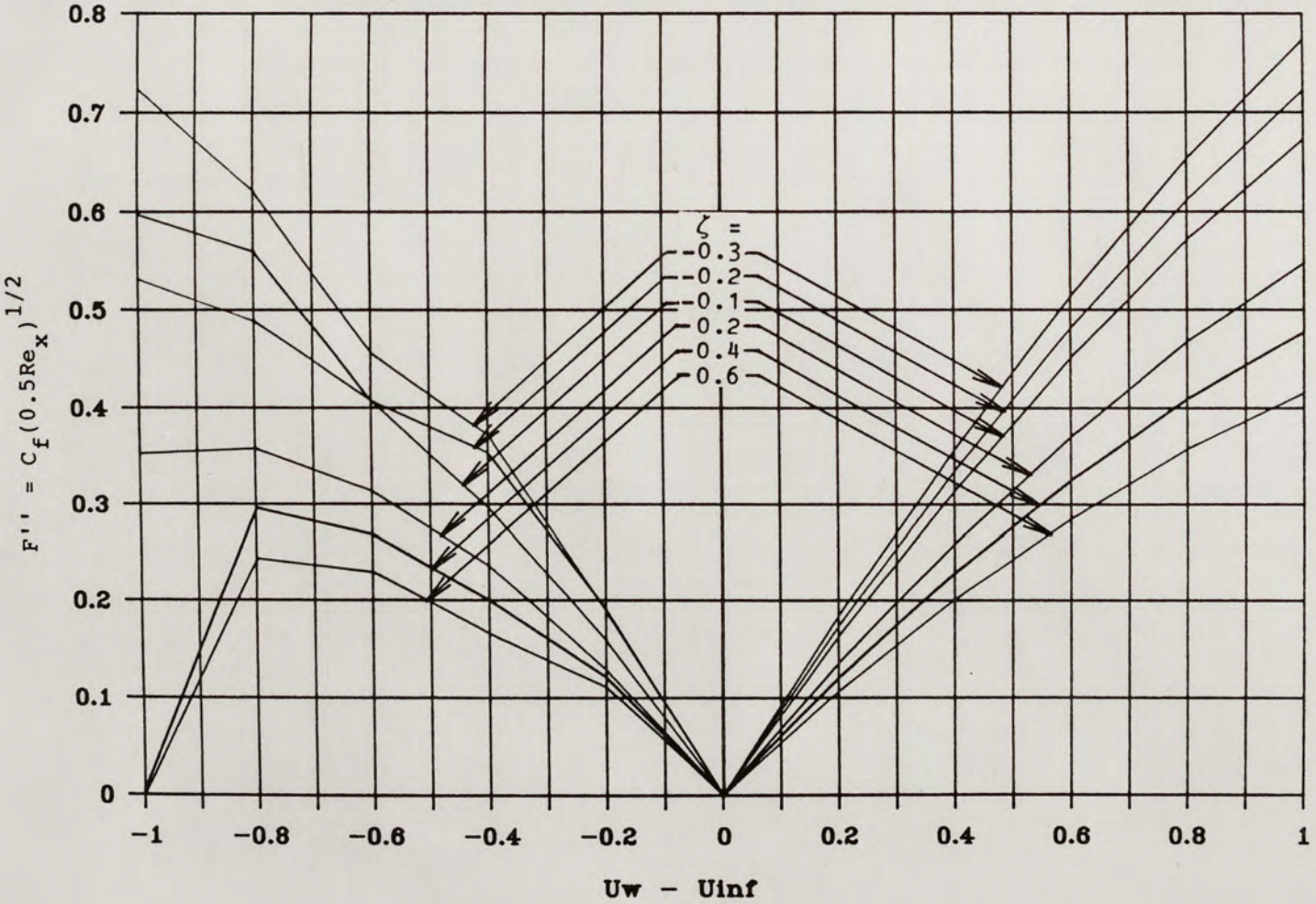

Figure 21. Friction Coefficient vs. ζ and Velocity Difference.

SUMMARY

From the analysis that has been performed, the velocity profiles show similar trends. In both the similar and non-similar boundary layer cases, as the velocity difference $|U_w - U_\infty|$ and relevant injection parameter, C or ζ, increase, the boundary layer thickness also increases. It was also observed that in both cases the shear stresses at the wall are greater when the wall velocity is greater than the free stream velocity, rather than when the free stream velocity is greater than the wall velocity while holding the velocity difference and injection parameter constant. Comparing the friction coefficient for the similar and non-similarity cases, it is seen that for the same values of the injection parameter and velocity difference $U_w - U_\infty$ the non-similarity is greater for BL suction and less for BL injection. It is also seen that separation for the non-similarity case occurs at an injection parameter, ζ, of about 0.4 which is approximately 35% less than for the corresponding similar case, where the injection parameter at separation is about 0.62.

For the solution of the energy equation, the similarity solutions showed that the Nusselt number was greatest for the case in which there was no velocity difference between

the plate and the free stream velocity. Also, the Nusselt number increased with increasing injection parameter, and decreased with increasing velocity difference. As was the case for the skin friction coefficient, the Nusselt number was greater for cases where $U_w > U_\infty$ than when $U_\infty > U_w$. Due to problems encountered in the solution of the energy equation in the non-similarity cases, no useful data were obtained.

Suggested areas for future investigation are 1) solve the energy equation for the non-similarity case and 2) mass transfer from the top of the boundary layer instead of through the bottom of the boundary layer. The latter suggestion has application for problems in two-phase flows.

APPENDICES

APPENDIX A

PROGRAM LISTING FOR SIMILAR BOUNDARY LAYER SOLUTION

```
D Line# 1      7                                   Microsoft FORTRAN77 V3.20 02/84
       1 $DEBUG
       2       PROGRAM VAUGHN
       3
       4 C     ***** THIS PROGRAM USES THE SECANT METHOD AND RUNGE-KUTTA
       5 C           METHOD TO CALCULATE THE SOLUTION OF BLASIUS' EQUATION
       6 C           FOR A MOVING FLAT PLATE WITH BOUNDARY LAYER INJECTION *****
       7
       8       REAL M, MI, NU
       9       INTEGER ULIMIT
      10       EXTERNAL FN
      11       DIMENSION CO(200,4), THETA(201), TABLE(54,161)
      12       COMMON /VECTOR/ ETA(201), F(201), P(201), Q(201)
      13       COMMON /BOUND/ CI
      14       OPEN(2,FILE='PRN')
      15       OPEN(10,FILE='C:VAUGHN.PRN',STATUS = 'NEW')
      16       WRITE(*,*) ' ENTER VALUE OF UPPER LIMIT
      17       READ(*,*) ULIMIT
      18       EPS = 1.0E-06
      19       WRITE(*,*) ' ENTER VALUE FOR PRANDTL NO.
      20       READ(*,*) PR
      21       H = 0.05
      22       INCR = IFIX(0.2/H)
      23       N = IFIX(10./H)
      24       NP1 = N + 1
      25       DO 2 I = 4, 54
1     26       TABLE(I,1) = FLOAT(I - 4)/5
1     27     2 CONTINUE
      28       DO 60 IDIF = ULIMIT, ULIMIT - 1, -1
1     29       DIFRNC = IDIF/10.
1     30       IFLAG = 0
1     31       UINF = 1.0
1     32       UWAL = 1.0 - DIFRNC
1     33    10 CONTINUE
1     34       DO 50 IC = 1, 10
1     35       GO TO (101, 102, 103, 104, 105, 106, 107, 108, 109, 110), IC
2     36   101 CI = -1.000
2     37       GO TO 15
2     38   102 CI = -0.750
2     39       GO TO 15
2     40   103 CI = -0.500
2     41       GO TO 15
2     42   104 CI = -0.250
2     43       GO TO 15
2     44   105 CI = 0.000
2     45       GO TO 15
2     46   106 CI = 0.125
2     47       GO TO 15
2     48   107 CI = 0.250
2     49       GO TO 15
2     50   108 CI = 0.375
2     51       GO TO 15
2     52   109 CI = 0.500
2     53       GO TO 15
2     54   110 CI = 0.619
2     55    15 CONTINUE
2     56       WRITE(10,2000) UINF, UWAL, CI
2     57       WRITE(*,2000) UINF, UWAL, CI
2     58       C = UINF
2     59       X1 = 0.1
```

```
D Line# 1      7                                   Microsoft FORTRAN77 V3.20 02/84
2     60       IF (UINF .LT. UWAL) X1 = -X1
2     61       CALL RUNGE(10.0, H, UWAL, X1, Y1, FN)
2     62       ITER = 1
2     63       X2 = 0.2
2     64       IF (UINF .LT. UWAL) X2 = -X2
2     65    20 CALL RUNGE(10.0, H, UWAL, X2, Y2, FN)
2     66       ITER = ITER + 1
2     67       MI = (Y2 - Y1) / (X2 - X1)
2     68       NU = C - Y2
2     69       IF (ABS(NU) .LT. EPS) GO TO 30
2     70       X1 = X2
2     71       X2 = X2 + NU / MI
2     72       IF ((X2 .LT. 0.0) .AND. (UINF .GT. UWAL)) X2 =  5.0E-05
2     73       IF ((X2 .GT. 0.0) .AND. (UINF .LT. UWAL)) X2 = -5.0E-05
2     74       Y1 = Y2
2     75       GO TO 20
2     76
2     77 C     ..... SOLVE THE ENERGY  EQUATION (3).  USE EQUATION (2.2.11)
2     78 C           FROM CHOW .....
2     79
2     80    30 CONTINUE
2     81
2     82       DO 40 I = 1, N
3     83       CO(I,1) = 1.0 - H*PR*F(I + 1)/4.
3     84       CO(I,2) = -2.0
3     85       CO(I,3) = 1.0 +H*PR*F(I + 1)/4.
3     86    40 CO(I,4) = -2.0*H*H*PR*Q(I)*Q(I)
2     87
2     88 C     ..... MODIFY COEFICIENTS ACCORDING TO (3.4.11 - 14) OF CHOW TO
2     89 C           INCORPORATE BOUNDARY CONDITIONS .....
2     90
2     91       THETA(NP1) = 0.0
2     92       CO(1,4) = CO(1,4) - CO(1,1)
2     93       CO(1,1) = 0.0
2     94       CO(N,4) = CO(N,4)
2     95       CO(N,3) = 0.0
2     96
2     97 C     ..... COMPUTE THETA(I) FOR I = 1,2,..., N. .....
2     98
2     99       CALL TRID(CO, THETA, N)
2    100
2    101       IUL = NP1 - 2
2    102       DO 45 I = 0, IUL
3    103       J = NP1 - I
3    104       THETA(J) = THETA(J - 1)
3    105    45 CONTINUE
2    106       THETA(1) = 1.0
2    107
2    108 C     ..... PRINT THE NUMBER OF ITERATIONS AND THEN PRINT F, F*, AND F**
2    109 C           AS FUNCTIONS OF ETA, WHERE AN * INDICATES DIFFERENTATION
2    110 C           WITH RESPECT TO ETA.  THUS F* IS F AND F** IS Q .....
2    111
2    112       THETAP = (THETA(1) - THETA(2))/H
2    113       WRITE(*,230) THETAP
2    114       WRITE(10,230) THETAP
2    115       WRITE(*,200) ITER
2    116       WRITE(10,200) ITER
2    117       WRITE(*,210) (ETA(I),F(I),P(I),Q(I),THETA(I), I = 1,N+4,INCR)
2    118       WRITE(10,210) (ETA(I),F(I),P(I),Q(I),THETA(I), I = 1,N+4,INCR)
```

```
D Line# 1      7
2   119        WRITE(*,220)
2   120        WRITE(10,220)
2   121     50 CONTINUE
1   122        IF (IFLAG .EQ. 1) THEN
1   123        IFLAG = 0
1   124        GO TO 60
1   125        ELSE
1   126        IFLAG = 1
1   127        UWAL = 1.0
1   128        UINF = 1.0 - DIFRNC
1   129        GO TO 10
1   130        ENDIF
1   131     60 CONTINUE
    132    200 FORMAT( 12X, I7 // 15X,3HETA, 10X,  1HF, 13X, 2HF*, 11X, 3HF**,
    133       A 10X, 5HTHETA / 14X, 5H-----, 4(5X, 9H---------) )
    134    210 FORMAT( 12X, F7.2, 4F14.7 )
    135    220 FORMAT(1X)
    136    230 FORMAT(12X, F10.7)
    137   2000 FORMAT(1H1,11X, F7.2, F14.2 / 11X, F8.3)
    138        CLOSE(UNIT=10)
    139        STOP
    140        END

Name     Type         Offset P Class

ABS                            INTRINSIC
C        REAL          38856
CI       REAL              0   /BOUND /
CO       REAL          35596
DIFRNC   REAL          38836
EPS      REAL          38800
ETA      REAL              0   /VECTOR/
F        REAL            804   /VECTOR/
FLOAT                          INTRINSIC
FN                             EXTERNAL
H        REAL          38808
I        INTEGER*4     38824
IC       INTEGER*4     38852
IDIF     INTEGER*4     38828
IFIX                           INTRINSIC
IFLAG    INTEGER*4     38840
INCR     INTEGER*4     38812
ITER     INTEGER*4     38868
IUL      INTEGER*4     38892
J        INTEGER*4     38900
M        REAL          *****
M1       REAL          38880
N        INTEGER*4     38816
NP1      INTEGER*4     38820
NU       REAL          38884
P        REAL           1608   /VECTOR/
PR       REAL          36804
Q        REAL           2412   /VECTOR/
TABLE    REAL            820
THETA    REAL             16
THETAP   REAL          38904
UINF     REAL          38844
ULIMIT   INTEGER*4     38796
UWAL     REAL          38848
```

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
X1      REAL            38860
X2      REAL            38872
Y1      REAL            38864
Y2      REAL            38876


     141
     142        FUNCTION FN(X, Y)
     143        FN = -X * Y / 2.
     144        RETURN
     145        END

Name     Type         Offset P Class

X        REAL              0 *
Y        REAL              4 *


     146
     147        SUBROUTINE RUNGE(ETAMAX, H, F0, G0, Y, FNCTN)
     148        COMMON /VECTOR/ ETA(201), F(201), P(201), G(201)
     149        COMMON /BOUND/ CI
     150
     151        I = 1
     152        ETA(I) = 0.0
     153        F(I) = -2.*CI
     154        P(I) = F0
     155        G(I) = G0
     156
     157 C      ..... APPLY RUNGE-KUTTA FORMULAE (9a - 9l) AND
     158 C            (10a - 10c) OF CHOW UNTIL ETAMAX IS REACHED .....
     159     10 D1F = H*P(I)
     160        D1P = H*G(I)
     161        D1G = H*FNCTN(F(I), G(I))
     162
     163        D2F = H*(P(I)+D1P/2.)
     164        D2P = H*(G(I)+D1G/2.)
     165        D2G = H*FNCTN(F(I) + D1F/2., G(I) + D1G/2.)
     166
     167        D3F = H*(P(I)+D2P/2.)
     168        D3P = H*(G(I)+D2G/2.)
     169        D3G = H*FNCTN(F(I) + D2F/2., G(I) + D2G/2.)
     170
     171        D4F = H*(P(I)+D3P)
     172        D4P = H*(G(I)+D3G)
     173        D4G = H*FNCTN(F(I) + D3F, G(I) + D3G)
     174
     175        F(I+1) = F(I) + (D1F + 2.*(D2F + D3F) + D4F)/6.0
     176        P(I+1) = P(I) + (D1P + 2.*(D2P + D3P) + D4P)/6.0
     177        G(I+1) = G(I) + (D1G + 2.*(D2G + D3G) + D4G)/6.0
     178        ETA(I+1) = ETA(I) + H
     179        I = I + 1
     180        IF( ETA(I)-ETAMAX ) 10, 20, 20
     181
     182 C      ..... KEEP ITERATING UNTIL THE APPROXIMATE BOUNDARY CONDITION
     183 C            (3.3.18) IS SATISFIED.  HALF-INTERVAL METHOD IS USED TO
     184 C            MODIFY THE STARTING VALUE OF G .....
     185
     186     20 Y = P(I)
```

```
D Line# 1      7                                  Microsoft FORTRAN-77 V3.20 02/84
     187       RETURN
     188       END

Name     Type          Offset P Class

CI       REAL               0    /BOUND /
D1F      REAL           39106
D1P      REAL           39110
D10      REAL           39114
D2F      REAL           39122
D2P      REAL           39126
D20      REAL           39130
D3F      REAL           39138
D3P      REAL           39142
D30      REAL           39146
D4F      REAL           39154
D4P      REAL           39158
D40      REAL           39162
ETA      REAL               0    /VECTOR/
ETAMAX   REAL               0 *
F        REAL             804    /VECTOR/
FNCTN                            SUB/FUN PARM
H        REAL               4 *
I        INTEGER*4      39102
P        REAL            1608    /VECTOR/
P0       REAL               8 *
Q        REAL            2412    /VECTOR/
Q0       REAL              12 *
Y        REAL              16 *


     189
     190         SUBROUTINE TRID(C, F, N)
     191 C       ***** SOLVING A SET OF LINEAR SIMULTANOUS EQUATIONS DESCRIBED
     192 C             BY (2.2.13) HAVING A TRIDIAGIONAL COEFICIENT MATRIX.
     193 C             COEFICIENT MATRICES ARE REPRESENTED BY C(I,J) AND THE
     194 C             SOLUTION IS STORED IN THE ONE-DIMENSIONAL ARRAY F(I) *****
     195
     196         DIMENSION C(200,4), F(201)
     197
     198 C       ***** ELIMINATE THE FIRST TERM OF EACH OF THE EQUATIONS (EXCEPT
     199 C             THE FIRST AND LAST ONE) ACCORDING TO (2.2.14) *****
     200         NM1 = N - 1
     201         DO 1 I = 2, NM1
1    202         C(I,2) = C(I,2) * C(I-1,2) - C(I,1) * C(I-1,3)
1    203         C(I,3) = C(I,3) * C(I-1,2)
1    204       1 C(I,4) = C(I,4) * C(I-1,2) - C(I,1) * C(I-1,4)
     205
     206 C       ***** COMPUTE SOLUTION USING (2.2.15) AND (2.2.16) *****
     207
     208         F(N) = ( C(N,4) * C(N-1,2) - C(N,1) * C(N-1,4) ) /
     209        A       ( C(N,2) * C(N-1,2) - C(N,1) * C(N-1,3) )
     210         DO 2 K = 1, NM1
1    211         J = N - K
1    212       2 F(J) = ( C(J,4) - C(J,3) * F(J+1) ) / C(J,2)
     213         RETURN
     214         END
```

D Line# 1 7

Name	Type	Offset	P	Class
C	REAL	0	*	
F	REAL	4	*	
I	INTEGER*4	39178		
J	INTEGER*4	39194		
K	INTEGER*4	39186		
N	INTEGER*4	8	*	
NM1	INTEGER*4	39174		

Name	Type	Size	Class
BOUND		4	COMMON
FN	REAL		FUNCTION
RUNGE			SUBROUTINE
TRID			SUBROUTINE
VAUGHN			PROGRAM
VECTOR		3216	COMMON

Pass One No Errors Detected
214 Source Lines

APPENDIX B

SAMPLE OUTPUT FOR SIMILAR BOUNDARY LAYER PROBLEM

FREESTREAM VELOCITY = 1.00 VELOCITY AT THE WALL = .00
INJECTION PARAMETER (C) = 0.000: PR = .72
NUMBER OF ITERATIONS = 6

ETA	F	F*	F**	THETA
0.00	0.0000000	.0000001	.3320573	.8477594
.20	.0066410	.0664078	.3319838	.8445846
.40	.0265599	.1327642	.3314698	.8350683
.60	.0597347	.1989373	.3300791	.8192528
.80	.1061083	.2647092	.3273892	.7972509
1.00	.1655718	.3297800	.3230071	.7692761
1.20	.2379487	.3937761	.3165892	.7356709
1.40	.3229816	.4562617	.3078654	.6969256
1.60	.4203208	.5167567	.2966634	.6536885
1.80	.5295181	.5747581	.2829310	.6067604
2.00	.6500244	.6297657	.2667515	.5570766
2.20	.7811934	.6813104	.2483509	.5056716
2.40	.9222901	.7289819	.2280917	.4536329
2.60	1.0725060	.7724550	.2064546	.4020444
2.80	1.2309774	.8115095	.1840066	.3519304
3.00	1.3968083	.8460444	.1613603	.3042013
3.20	1.5690949	.8760813	.1391280	.2596107
3.40	1.7469500	.9017611	.1178762	.2187260
3.60	1.9295250	.9233295	.0980863	.1819172
3.80	2.1160297	.9411178	.0801259	.1493605
4.00	2.3057461	.9555181	.0642341	.1210567
4.20	2.4980392	.9669569	.0505197	.0968588
4.40	2.6923604	.9758707	.0389726	.0765055
4.60	2.8882475	.9826833	.0294838	.0596558
4.80	3.0853202	.9877893	.0218712	.0459216
5.00	3.2832732	.9915417	.0159068	.0348962
5.20	3.4818671	.9942454	.0113418	.0261770
5.40	3.6809185	.9961551	.0079277	.0193833
5.60	3.8802900	.9974776	.0054320	.0141670
5.80	4.0798812	.9983753	.0036484	.0102198
6.00	4.2796197	.9989727	.0024020	.0072761
6.20	4.4794559	.9993624	.0015502	.0051123
6.40	4.6793551	.9996115	.0009806	.0035446
6.60	4.8792944	.9997677	.0006080	.0024250
6.80	5.0792584	.9998636	.0003696	.0016370
7.00	5.2792373	.9999213	.0002202	.0010902
7.20	5.4792256	.9999554	.0001286	.0007163
7.40	5.6792188	.9999751	.0000736	.0004642
7.60	5.8792152	.9999864	.0000413	.0002967
7.80	6.0792131	.9999926	.0000227	.0001870
8.00	6.2792120	.9999960	.0000122	.0001162
8.20	6.4792109	.9999979	.0000065	.0000712
8.40	6.6792097	.9999988	.0000034	.0000429
8.60	6.8792095	.9999993	.0000017	.0000254
8.80	7.0792103	.9999996	.0000008	.0000148
9.00	7.2792110	.9999997	.0000004	.0000084
9.20	7.4792118	.9999997	.0000002	.0000046
9.40	7.6792126	.9999997	.0000001	.0000024
9.60	7.8792133	.9999997	.0000000	.0000011
9.80	8.0792141	.9999997	.0000000	.0000004
10.00	8.2792149	.9999997	.0000000	0.0000000

APPENDIX C

PROGRAM LISTING FOR NON-SIMILAR BOUNDARY LAYER SOLUTION

```
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
       1 $DEBUG
       2       PROGRAM CONTHS
       3
       4 C     ***** THIS PROGRAM USES THE SECANT METHOD AND RUNGE-KUTTA
       5 C           METHOD TO CALCULATE THE SOLUTION OF BLASIUS' EQUATION
       6 C           FOR A MOVING FLAT PLATE WITH BOUNDARY LAYER INJECTION *****
       7
       8       IMPLICIT REAL*8 (A-G,O-Z)
       9       REAL*8 MIF, MIG, NUF, NUG
      10       INTEGER ULIMIT
      11       EXTERNAL FF, FG
      12       DIMENSION BEST(4)
      13       COMMON /VECTOR/ ETA(201), F(201), P(201), Q(201),
      14      1                          G(201), R(201), S(201)
      15       COMMON /BOUND/ ZETA
      16       COMMON /PRANTL/ PR
      17       COMMON ITRUBL
      18       DATA ITMAX1, ITMAX2/15, 150/
      19       OPEN(2,FILE='PRN')
      20       OPEN(10,FILE = 'CONTHS.PRN', STATUS = 'NEW')
      21       DO 5 I = 1, 201
1     22       ETA(I) = 0.0
1     23       F(I)   = 0.0
1     24       P(I)   = 0.0
1     25       Q(I)   = 0.0
1     26       G(I)   = 0.0
1     27       R(I)   = 0.0
1     28       S(I)   = 0.0
1     29     5 CONTINUE
      30       WRITE(*,*) '  ENTER VALUE OF UPPER LIMIT '
      31       READ(*,*) ULIMIT
      32       ETAMAX = 6.0
      33       EPS = 5.0E-06
      34       WRITE(*,*) ' ENTER A VALUE FOR THE PRANDTL NUMBER.'
      35       READ(*,*) PR
      36       DO 110 IDIF = ULIMIT, ULIMIT - 2, -2
1     37       DIFRNC = IDIF/10.
1     38       IFLAG = 0
1     39       UINF = 1.0
1     40       UWAL = 1.0 - DIFRNC
1     41    10 CONTINUE
1     42       DO 100 IC = 1, 7
2     43       XF1 = 0.6
2     44       XG1 = -0.7
2     45       XT1 = -0.5
2     46       XP1 = 0.1
2     47       GO TO (114, 115, 116, 117, 118, 119, 120), IC
2     48   114 ZETA = -0.3
2     49       H = 0.05
2     50       INCR = IFIX(0.2/H)
2     51       N = IFIX(6.0/H)
2     52       NP1 = N + 1
2     53       GO TO 15
2     54   115 ZETA = -0.2
2     55       H = 0.05
2     56       INCR = IFIX(0.2/H)
2     57       N = IFIX(6.0/H)
2     58       NP1 = N + 1
2     59       GO TO 15
```

```
2    60   116 ZETA = -0.1
2    61       H = 0.1
2    62       INCR = IFIX(0.2/H)
2    63       N = IFIX(6.0/H)
2    64       NP1 = N + 1
2    65       GO TO 15
2    66   117 ZETA = 0.0
2    67       H = 0.1
2    68       INCR = IFIX(0.2/H)
2    69       N = IFIX(6.0/H)
2    70       NP1 = N + 1
2    71       GO TO 15
2    72   118 ZETA = 0.2
2    73       H = 0.1
2    74       INCR = IFIX(0.2/H)
2    75       N = IFIX(6.0/H)
2    76       NP1 = N + 1
2    77       GO TO 15
2    78   119 ZETA = 0.4
2    79       H = 0.05
2    80       INCR = IFIX(0.2/H)
2    81       N = IFIX(6.0/H)
2    82       NP1 = N + 1
2    83       GO TO 15
2    84   120 ZETA = 0.6
2    85       H = 0.05
2    86       INCR = IFIX(0.2/H)
2    87       N = IFIX(6.0/H)
2    88       NP1 = N + 1
2    89    15 CONTINUE
2    90       WRITE(2,2000) UINF, UWAL, ZETA, PR
2    91       WRITE(*,2000) UINF, UWAL, ZETA, PR
2    92       CONST = UINF
2    93       IF (UINF .LT. UWAL) XF1 = -ABS(XF1)
2    94       CALL RUNGE(ETAMAX, H, UWAL, XF1, XG1, YF1, YG1, FF, FG)
2    95       ERRORV = ABS(CONST - YF2) + ABS(YG2)
2    96       BEST(1) = XF1
2    97       BEST(2) = XG1
2    98       BEST(3) = CONST - YF1
2    99       BEST(4) = -YG1
2   100       ITER = 1
2   101       ITERR = 1
2   102       XF2 = 0.9*XF1
2   103       XG2 = 0.9*XG1
2   104    20 CONTINUE
2   105       CALL RUNGE(ETAMAX, H, UWAL, XF2, XG2, YF2, YG2, FF, FG)
2   106       WRITE(*,300) XF2, YF2, XG2, YG2
2   107       ITER = ITER + 1
2   108       ITERR = ITERR + 1
2   109       IF (ITERR .GE. ITMAX1) THEN
2   110       ITERR = 0
2   111       XF2 = XF2 + 0.01
2   112       XG2 = XG2 + 0.01
2   113       GO TO 30
2   114       ENDIF
2   115       IF (ITRUBL .EQ. 1) THEN
2   116       ITRUBL = 0
2   117       XF2 = 0.5*(XF2 + XF1)
2   118       XG2 = 0.5*(XG2 + XG1)
```

```
D Line# 1      7                                    Microsoft FORTRAN77 V3.20 02/84
2    119       ITERR = 0
2    120       WRITE(*,*)  ' RETURN A
2    121       GO TO 40
2    122       ENDIF
2    123       MIF = (YF2 - YF1) / (XF2 - XF1)
2    124       NUF = CONST - YF2
2    125       NUG = -YG2
2    126       XF1 = XF2
2    127       YF1 = YF2
2    128       XF2 = XF2 + NUF / MIF
2    129       IF ((ABS(NUF) .LT. EPS) .AND. (ABS(NUG) .LT. EPS)) GO TO 50
2    130       IF (ERRORV .GE. ABS(NUF) + ABS(NUG)) THEN
2    131       ERRORV = ABS(NUF) + ABS(NUG)
2    132       BEST(1) = XF2
2    133       BEST(2) = XG2
2    134       BEST(3) = NUF
2    135       BEST(4) = NUG
2    136       ENDIF
2    137       IF (ABS(NUF) .LT. EPS) THEN
2    138       ITERR = 0
2    139       WRITE(*,*)  ' RETURN B
2    140       GO TO 30
2    141       ENDIF
2    142       IF (ITMAX1 .EQ. ITERR) THEN
2    143       ITERR = 0
2    144       WRITE(*,*)  ' RETURN C '
2    145       GO TO 30
2    146       ENDIF
2    147       IF (ITER .GT. ITMAX2) THEN
2    148       GO TO 40
2    149       ELSE
2    150       GO TO 20
2    151       ENDIF
2    152    30 CONTINUE
2    153       CALL RUNGE(ETAMAX, H, UWAL, XF2, XG2, YF2, YG2, FF, FG)
2    154       WRITE(*,300) XF2, YF2, XG2, YG2
2    155       ITER = ITER + 1
2    156       ITERR = ITERR + 1
2    157       IF ((YG2 .EQ. YG1) .OR. (XG2 .EQ. XG1)) THEN
2    158       ITERR = 0
2    159       WRITE(*,*)  ' RETURN 1
2    160       GO TO 40
2    161       ENDIF
2    162       MIG = (YG2 - YG1) / (XG2 - XG1)
2    163       NUF = CONST - YF2
2    164       NUG = -YG2
2    165       XG1 = XG2
2    166       YG1 = YG2
2    167       XG2 = XG2 + NUG / MIG
2    168       IF (XG2 .GT. XG1 + 2.0) XG2 = XG1 + 2.0
2    169       IF ((ABS(NUF) .LT. EPS) .AND. (ABS(NUG) .LT. EPS)) GO TO 50
2    170       IF (ERRORV .GE. ABS(NUF) + ABS(NUG)) THEN
2    171       ERRORV = ABS(NUF) + ABS(NUG)
2    172       BEST(1) = XF2
2    173       BEST(2) = XG2
2    174       BEST(3) = NUF
2    175       BEST(4) = NUG
2    176       ENDIF
2    177       IF (ABS(NUG) .LT. EPS) THEN
```

```
D Line# 1      7                                    Microsoft FORTRAN77 V3.20 02/84
2   178        BEST(4) = NUG
2   179        ENDIF
2   180        IF (ABS(NUG) .LT. EPS) THEN
2   181        ITERR = 0
2   182        WRITE(*,*)    RETURN 2 '
2   183        GO TO 20
2   184        ENDIF
2   185        IF (ITERR .EQ. ITMAX1) THEN
2   186        ITERR = 0
2   187        WRITE(*,*)    RETURN 3 '
2   188        GO TO 20
2   189        ENDIF
2   190        IF (ITER .GT. ITMAX2) THEN
2   191        ITERR = 0
2   192        GO TO 40
2   193        ELSE
2   194        GO TO 30
2   195        ENDIF
2   196     40 CONTINUE
2   197        WRITE(*,240) BEST(3), BEST(4)
2   198        WRITE(2,240) BEST(3), BEST(4)
2   199        CALL RUNGE(ETAMAX, H, UWAL, BEST(1), BEST(2), YF2, YG2, FF, FG)
2   200     50 CONTINUE
2   201
2   202        WRITE(*,200)
2   203        WRITE(2,200)
2   204        WRITE(*,210)  (ETA(I),F(I),P(I),Q(I), I=1,N+INCR,INCR)
2   205        WRITE(2,210)  (ETA(I),F(I),P(I),Q(I), I=1,N+INCR,INCR)
2   206        WRITE(10,210) (ETA(I),F(I),P(I),Q(I), I=1,N+INCR,INCR)
2   207        WRITE(*,220)
2   208        WRITE(2,220)
2   209        XF2 = 0.9*XF2
2   210        XG2 = 0.9*XG2
2   211        XT2 = 0.9*XT2
2   212        XP2 = 0.9*XP2
2   213
2   214    100 CONTINUE
1   215        IF (IFLAG .EQ. 1) THEN
1   216        IFLAG = 0
1   217        GO TO 110
1   218        ELSE
1   219        IFLAG = 1
1   220        UWAL = 1.0
1   221        UINF = 1.0 - DIFRNC
1   222        GO TO 10
1   223        ENDIF
1   224    110 CONTINUE
    225
    226        CLOSE(10)
    227
    228    200 FORMAT( 1H0, 15X,3HETA, 10X, 1HF, 13X, 2HF*, 11X, 3HF** / 14X,
    229       A         5H-----, 3(5X, 9H---------) )
    230    210 FORMAT( 12X, F7.2, 3F14.7 )
    231    220 FORMAT(1X)
    232    230 FORMAT(12X, F10.7)
    233    240 FORMAT(1H0, 11X, 'CONVERGENCE DID NOT OCCUR.' / 12X,
    234       A         'ERROR FOR F =', F10.6/ 12X, 'ERROR FOR G =', F10.6)
    235    250 FORMAT(1H0, 11X, 'CONVERGENCE DID NOT OCCUR.' / 12X,
    236       A       'ERROR FOR THETA =', F10.6/ 12X, 'ERROR FOR PHI =', F10.6)
```

```
                                                                    Page   5
                                                                    03-24-88
                                                                    09:10:23
D Line# 1      7                                  Microsoft FORTRAN77 V3.20 02/84
     237        A              /P2 =', F10.5)
     238   2000 FORMAT(1H1,10X, ' FREESTREAM VELOCITY =', F7.2, ' VELOCITY AT THE
     239        A WALL =', F7.2 /, 12X, 'INJECTION PARAMETER (ZETA) =', F8.3)
     240         STOP
     241         END

Name     Type          Offset P Class

ABS                             INTRINSIC
BEST     REAL*8           20
CONST    REAL*8          172
DIFRNC   REAL*8           92
EPS      REAL*8           76
ERRORV   REAL*8          196
ETA      REAL*8            0   /VECTOR/
ETAMAX   REAL*8           68
F        REAL*8         1608   /VECTOR/
FF                              EXTERNAL
FG                              EXTERNAL
G        REAL*8         6432   /VECTOR/
H        REAL            156
I        INTEGER*4        60
IC       INTEGER*4       120
IDIF     INTEGER*4        84
IFIX                            INTRINSIC
IFLAG    INTEGER*4       100
INCR     INTEGER*4       160
ITER     INTEGER*4       220
ITERR    INTEGER*4       224
ITMAX1   INTEGER*4        52
ITMAX2   INTEGER*4        56
ITRUBL   INTEGER*4         0   /COMMQQ/
MIF      REAL*8          244
MIG      REAL*8          268
N        INTEGER*4       164
NP1      INTEGER*4       168
NUF      REAL*8          252
NUG      REAL*8          260
P        REAL*8         3216   /VECTOR/
Q        REAL*8         4824   /VECTOR/
R        REAL*8         8040   /VECTOR/
S        REAL*8         9648   /VECTOR/
UINF     REAL*8          104
ULIMIT   INTEGER*4        64
UWAL     REAL*8          112
XF1      REAL*8          124
XF2      REAL*8          228
XG1      REAL*8          132
XG2      REAL*8          236
XP1      REAL*8          148
XP2      REAL*8          292
XT1      REAL*8          140
XT2      REAL*8          284
YF1      REAL*8          180
YF2      REAL*8          204
YG1      REAL*8          188
YG2      REAL*8          212
ZETA     REAL*8            0   /BOUND /
     242
```

```
D Line# 1      7                                    Microsoft FORTRAN77 V3.20 02/84
     243       FUNCTION FF(V, W, X, Y, Z)
     244       IMPLICIT REAL*8 (A-G,O-Z)
     245       COMMON /BOUND/ ZETA
     246       COMMON ITRUBL
     247       FF = ZETA*(X*Z - Y*W) - V*Y
     248       IF (ABS(FF) .LT. 100.0) GO TO 10
     249       FF = 0.0
     250       ITRUBL = 1
     251    10 CONTINUE
     252       RETURN
     253       END

Name     Type         Offset P Class

ABS                             INTRINSIC
ITRUBL  INTEGER*4          0    /COMMOO/
V       REAL*8             0 *
W       REAL*8             4 *
X       REAL*8             8 *
Y       REAL*8            12 *
Z       REAL*8            16 *
ZETA    REAL*8             0    /BOUND /


     254
     255       FUNCTION FG(U, V, W, X, Y, Z)
     256       IMPLICIT REAL*8 (A-G,O-Z)
     257       FG = W*Y - U*Z - 2.0*X*V
     258       RETURN
     259       END

Name     Type         Offset P Class

U       REAL*8             0 *
V       REAL*8             4 *
W       REAL*8             8 *
X       REAL*8            12 *
Y       REAL*8            16 *
Z       REAL*8            20 *


     260
     261       SUBROUTINE RUNGE(ETAMAX, H, P0, Q0, S0, YF, YG, FNC, GNC)
     262       IMPLICIT REAL*8 (A-G,O-Z)
     263       COMMON /VECTOR/ ETA(201), F(201), P(201), Q(201),
     264      1                          G(201), R(201), S(201)
     265       COMMON /BOUND/ ZETA
     266       COMMON ITRUBL
     267
     268       I = 1
     269       ETA(I) = 0.0
     270       F(I) = - ZETA/2.0
     271       P(I) = P0
     272       Q(I) = Q0
     273       G(I) = -0.5
     274       R(I) = 0.0
     275       S(I) = S0
     276
     277 C     ..... APPLY RUNGE-KUTTA FORMULAE (29a - 29x) AND
```

D Line# 1 7 Microsoft FORTRAN77 V3.20 02/84

```
C            (30a through 30f) UNTIL ETAMAX IS REACHED .....
   10 D1F = H*F(I)
      D1P = H*Q(I)
      D1Q = H*FNC(F(I), G(I), P(I), Q(I), R(I))
      D1G = H*R(I)
      D1R = H*S(I)
      D1S = H*GNC(F(I), G(I), P(I), Q(I), R(I), S(I))

      D2F = H*(P(I) + D1P/2.)
      D2P = H*(Q(I) + D1Q/2.)
      D2Q = H*FNC(F(I) + D1F/2., G(I) + D1G/2., P(I) + D1P/2.,
     A            Q(I) + D1Q/2., R(I) + D1R/2.)
      D2G = H*(R(I) + D1R/2.)
      D2R = H*(S(I) + D1S/2.)
      D2S = H*GNC(F(I) + D1F/2., G(I) + D1G/2., P(I) + D1P/2.,
     A            Q(I) + D1Q/2., R(I) + D1R/2., S(I) + D1S/2.)

      D3F = H*(P(I) + D2P/2.)
      D3P = H*(Q(I) + D2Q/2.)
      D3Q = H*FNC(F(I) + D2F/2., G(I) + D2G/2., P(I) + D2P/2.,
     A            Q(I) + D2Q/2., R(I) + D2R/2.)
      D3G = H*(R(I) + D2R/2.)
      D3R = H*(S(I) + D2S/2.)
      D3S = H*GNC(F(I) + D2F/2., G(I) + D2G/2., P(I) + D2P/2.,
     A            Q(I) + D2Q/2., R(I) + D2R/2., S(I) + D2S/2.)

      D4F = H*(P(I) + D3P)
      D4P = H*(Q(I) + D3Q)
      D4Q = H*FNC(F(I) + D3F, G(I) + D3G, P(I) + D3P,
     A            Q(I) + D3Q, R(I) + D3R)
      D4G = H*(R(I) + D3R)
      D4R = H*(S(I) + D3S)
      D4S = H*GNC(F(I) + D3F, G(I) + D3G, P(I) + D3P,
     A            Q(I) + D3Q, R(I) + D3R, S(I) + D3S)

      F(I+1) = F(I) + (D1F + 2.*(D2F + D3F) + D4F)/6.0
      P(I+1) = P(I) + (D1P + 2.*(D2P + D3P) + D4P)/6.0
      Q(I+1) = Q(I) + (D1Q + 2.*(D2Q + D3Q) + D4Q)/6.0
      G(I+1) = G(I) + (D1G + 2.*(D2G + D3G) + D4G)/6.0
      R(I+1) = R(I) + (D1R + 2.*(D2R + D3R) + D4R)/6.0
      S(I+1) = S(I) + (D1S + 2.*(D2S + D3S) + D4S)/6.0

      IF (ITRUBL .EQ. 1) GO TO 20
      ETA(I+1) = ETA(I) + H
      I = I + 1
      IF( ETA(I)-ETAMAX ) 10, 20, 20

C     ..... KEEP ITERATING UNTIL THE APPROXIMATE BOUNDARY CONDITION
C           (3.3.18) IS SATISFIED.  HALF-INTERVAL METHOD IS USED TO
C           MODIFY THE STARTING VALUE OF Q .....

   20 YF = P(I)
      YG = R(I)
      RETURN
      END
```

Name	Type	Offset	P	Class
D1F	REAL*8	980		

```
D Line# 1      7
D1G      REAL*8          1012
D1P      REAL*8           988
D1Q      REAL*8           996
D1R      REAL*8          1020
D1S      REAL*8          1028
D2F      REAL*8          1044
D2G      REAL*8          1076
D2P      REAL*8          1052
D2Q      REAL*8          1060
D2R      REAL*8          1084
D2S      REAL*8          1092
D3F      REAL*8          1108
D3G      REAL*8          1140
D3P      REAL*8          1116
D3Q      REAL*8          1124
D3R      REAL*8          1148
D3S      REAL*8          1156
D4F      REAL*8          1172
D4G      REAL*8          1204
D4P      REAL*8          1180
D4Q      REAL*8          1188
D4R      REAL*8          1212
D4S      REAL*8          1220
ETA      REAL*8             0    /VECTOR/
ETAMAX   REAL*8             0 *
F        REAL*8          1608    /VECTOR/
FNC                              SUB/FUN PARM
G        REAL*8          6432    /VECTOR/
GNC                              SUB/FUN PARM
H        REAL               4 *
I        INTEGER*4        976
ITRUBL   INTEGER*4          0    /COMMQQ/
P        REAL*8          3216    /VECTOR/
PQ       REAL*8             8 *
Q        REAL*8          4824    /VECTOR/
QQ       REAL*8            12 *
R        REAL*8          8040    /VECTOR/
S        REAL*8          9648    /VECTOR/
SQ       REAL*8            16 *
YF       REAL*8            20 *
YG       REAL*8            24 *
ZETA     REAL*8             0    /BOUND /

    333

Name     Type            Size    Class

BOUND                       8    COMMON
COMMQQ                      4    COMMON
CONTHS                           PROGRAM
FF       REAL*8                  FUNCTION
FG       REAL*8                  FUNCTION
RUNGE                            SUBROUTINE
VECTOR                  11256    COMMON

 Pass One     No Errors Detected
             333 Source Lines
```

APPENDIX D

SAMPLE OUTPUT FOR NON-SIMILAR BOUNDARY LAYER PROBLEM

FREESTREAM VELOCITY = 1.00 VELOCITY AT THE WALL = .00
INJECTION PARAMETER (ZETA) = .000

ETA	F	F*	F**
.00	.0000000	.0000000	.4696005
.20	.0093915	.0939054	.4693066
.40	.0375493	.1876054	.4672547
.60	.0843858	.2805758	.4617350
.80	.1496748	.3719637	.4511905
1.00	.2329905	.4606331	.4343796
1.20	.3336580	.5452471	.4105658
1.40	.4507243	.6243869	.3796926
1.60	.5829572	.6967002	.3424874
1.80	.7288733	.7610580	.3004456
2.00	.8867980	.8166953	.2556694
2.20	1.0549485	.8633049	.2105801
2.40	1.2315292	.9010661	.1675605
2.60	1.4148259	.9306019	.1286130
2.80	1.6032855	.9528761	.0951137
3.00	1.7955700	.9690552	.0677107
3.20	1.9905832	.9803655	.0463708
3.40	2.1874697	.9879710	.0305358
3.60	2.3855930	.9928884	.0193293
3.80	2.5845015	.9959448	.0117593
4.00	2.7838893	.9977706	.0068746
4.20	2.9835583	.9988189	.0038618
4.40	3.1833859	.9993972	.0020845
4.60	3.3832993	.9997038	.0010810
4.80	3.5832575	.9998601	.0005387
5.00	3.7832381	.9999365	.0002579
5.20	3.9832295	.9999725	.0001187
5.40	4.1832258	.9999888	.0000524
5.60	4.3832244	.9999959	.0000223
5.80	4.5832239	.9999988	.0000091
6.00	4.7832238	1.0000000	.0000036

REFERENCES

1. Abdelhafez, Talat A. "Skin Friction and Heat Transfer on a Continuous Flat Surface Moving in a Parallel Free Stream." International Heat and Mass Transfer 28 (1985): 1234-1237.

2. Bradbury, T. C. Theoretical Mechanics. New York: John Wiley & Sons, 1968.

3. Schlichting, Dr. Herman. Boundary Layer Theory. 7th ed. New York: McGraw-Hill Book Company, 1979.

4. Burmeister, Louis C. Convective Heat Transfer. New York: John Wiley & Sons, 1983.

5. Chow, Chuen-Yen. An Introduction to Computational Fluid Mechanics. Boulder, Colorado: Seminole Publishing Company, 1983.

6. Zucrow, Maurice and Joe D. Hoffman. Gas Dynamics. Vol. 1. New York: John Wiley & Sons, Inc., 1976.

7. Sparrow, E. M., H. Quack and C. J. Boerner. "Local Non-Similarity Boundary Layer Solutions." American Institute of Aeronautics and Astronautics 8 (November 1979): 1936-1942.