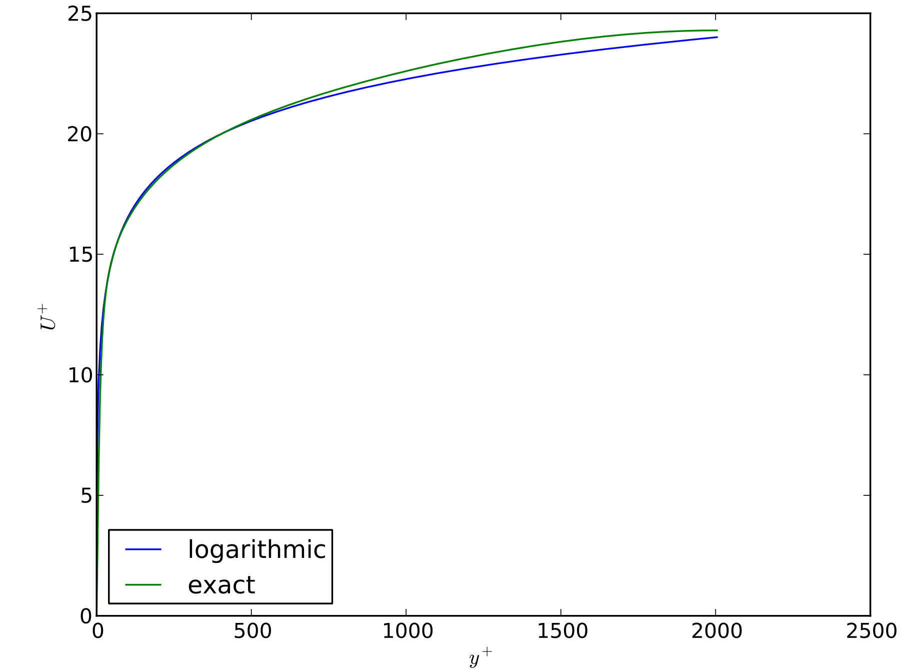
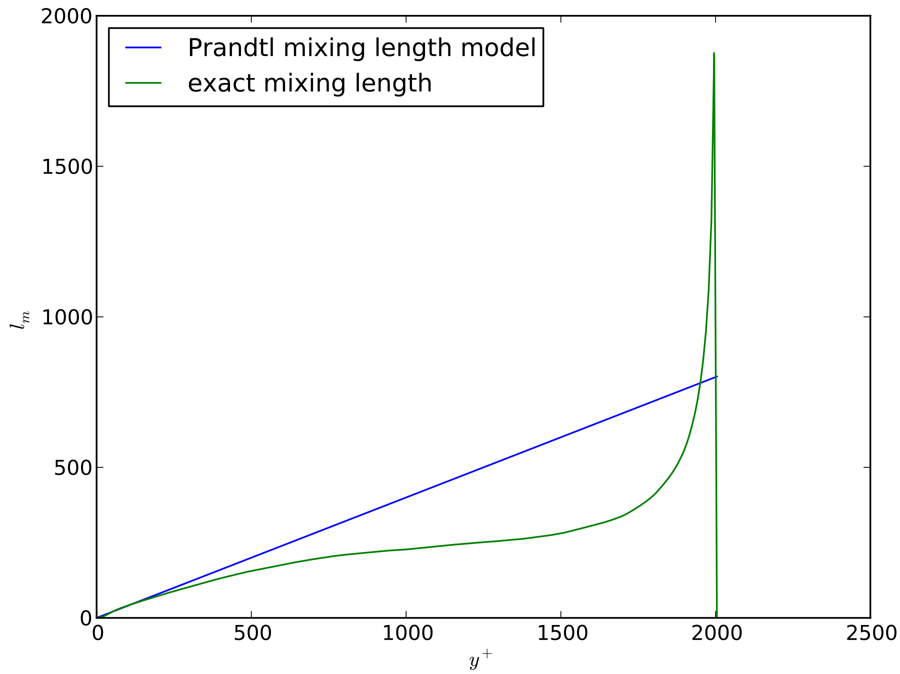
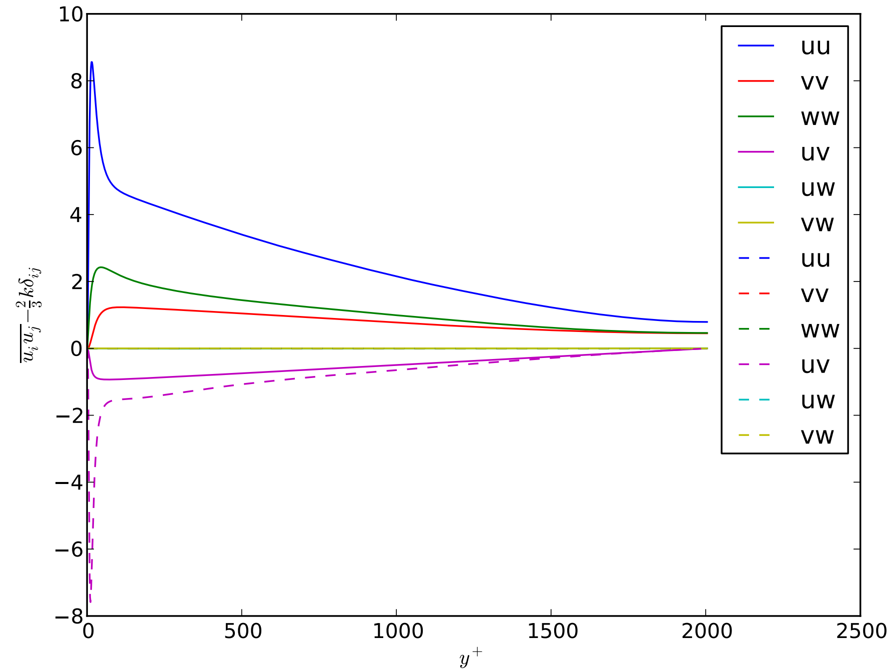


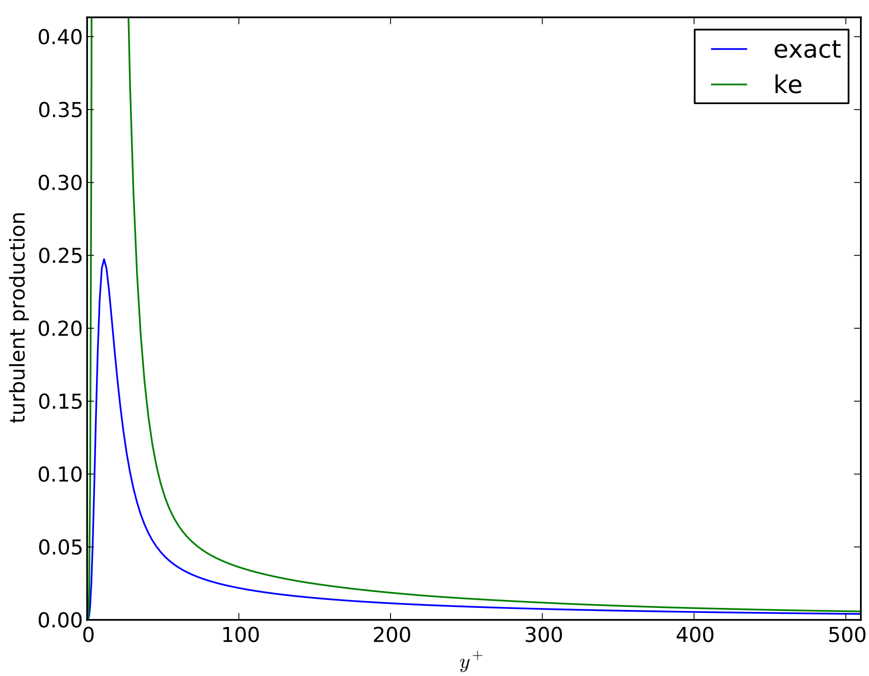
25
20
15
10
5
0
0
500
1000
1500
2000
2500
U^+
y^+
logarithmic
exact



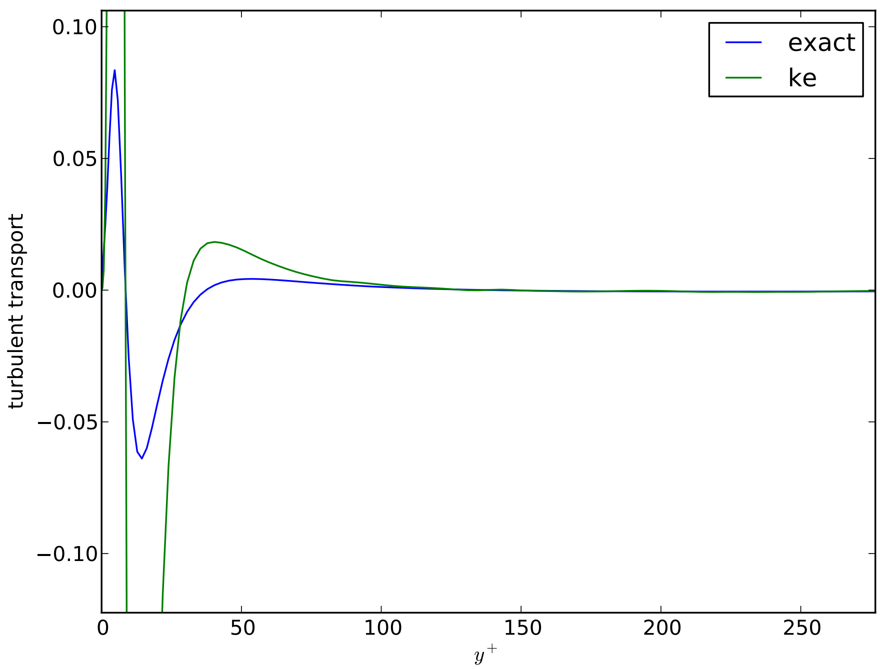
2000
1500
1000
500
0
0
500
1000
1500
2000
2500
l_m
y^+
Prandtl mixing length model
exact mixing length



10
8
6
4
2
0
−2
−4
−6
−8
0
500
1000
1500
2000
2500
$\overline{u_i u_j} - \frac{2}{3} k \delta_{ij}$
y^+
uu
vv
ww
uv
uw
vw
uu
vv
ww
uv
uw
vw



0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
100
200
300
400
500
turbulent production
y^+
exact
ke


exact
ke
turbulent transport
0.10
0.05
0.00
−0.05
−0.10
0
50
100
150
200
250
y^+

```
from pylab import *

close('all')

# function to calculate numerical derivatives
def numericalDerivative(u):
  dudy = zeros(N)
  for i in range(0,N):
    if i > 0 and i < N-1:
      dym = yp[i+1]-yp[i]
      dyp = yp[i]-yp[i-1]
      dudy[i] = u[i+1]*dym/((dym+dyp)*dyp) \
          + u[i]*(dyp-dym)/(dyp*dym) \
          - u[i-1]*dyp/((dym+dyp)*dym)
    elif i > 0:
      dy = yp[i]-yp[i-1]
      dudy[i] = (u[i]-u[i-1])/dy
    elif i < N-1:
      dy = yp[i+1]-yp[i]
      dudy[i] = (u[i+1]-u[i])/dy

  return dudy

# Simulation parameters
Re = 2003
nu = 1/0.485e+5
u_tau = 0.41302030e-1
delta = Re*nu/u_tau
kappa =0.4

prof = loadtxt('profiles/Re2000.prof', comments='%')
bal = loadtxt('balances/Re2000.bal.kbal', comments='%')

yt = prof[:,0]
yp = prof[:,1]
Up = prof[:,2]
up = prof[:,3]
vp = prof[:,4]
wp = prof[:,5]
uvp = prof[:,10]
uwp = prof[:,11]
vwp = prof[:,12]

epsilon = -bal[:,2]
production = bal[:,3]
p_diff = bal[:,5]
t_diff = bal[:,6]

k = 0.5*(up*up+vp*vp+wp*wp)

N = size(yp)

# Check log profile
figure()
plot(yp[1:], log(yp[1:])/kappa+5, label='logarithmic')
plot(yp, Up, label='exact')
legend(loc='best')
xlabel(r'$y^+$')
ylabel(r'$U^+$')

# Check mixing length validity
dUdy = numericalDerivative(Up)
dkdy = numericalDerivative(k)
```

```
l_m = kappa*delta*yp
l_exact = sqrt(abs(uvp/dUdy**2))

figure()
plot(yp, l_m, label='Prandtl mixing length model')
plot(yp, l_exact, label='exact mixing length')
legend(loc='best')
xlabel(r'$y^+$')
ylabel(r'$l_m$')

# Compare ke and exact Reynolds stress predictions
uu_exact = up*up-2/3*k
vv_exact = vp*vp-2/3*k
ww_exact = wp*wp-2/3*k
uv_exact = uvp
uw_exact = uwp
vw_exact = vwp

nu_t_ke = 0.09*k**2/epsilon
uu_ke = zeros(N)
vv_ke = zeros(N)
ww_ke = zeros(N)
uv_ke = -nu_t_ke*dUdy
uw_ke = zeros(N)
vw_ke = zeros(N)

figure()
plot(yp, uu_exact, label='uu', color='b')
plot(yp, vv_exact, label='vv', color='r')
plot(yp, ww_exact, label='ww', color='g')
plot(yp, uv_exact, label='uv', color='m')
plot(yp, uw_exact, label='uw', color='c')
plot(yp, vw_exact, label='vw', color='y')

plot(yp, uu_ke, linestyle='--', label='uu', color='b')
plot(yp, vv_ke, linestyle='--', label='vv', color='r')
plot(yp, ww_ke, linestyle='--', label='ww', color='g')
plot(yp, uv_ke, linestyle='--', label='uv', color='m')
plot(yp, uw_ke, linestyle='--', label='uw', color='c')
plot(yp, vw_ke, linestyle='--', label='vw', color='y')
legend(loc='best')
xlabel(r'$y^+$')
ylabel(r'$\overline{u_iu_j}-\frac{2}{3}k\delta_{ij}$')

# ke estimate of the production term
production_ke = nu_t_ke*dUdy**2

figure()
plot(yp, production, label='exact')
plot(yp, production_ke, label='ke')
legend(loc='best')
xlabel(r'$y^+$')
ylabel(r'turbulent production')

figure()
transport_exact = t_diff + p_diff
transport_ke = numericalDerivative(nu_t_ke*dkdy)
plot(yp, transport_exact, label='exact')
plot(yp, transport_ke, label='ke')
legend(loc='best')
xlabel(r'$y^+$')
ylabel(r'turbulent transport')

show()
```