ABSTRACT

HOLLAND, ANDREW P. Developing A Model To Estimate Tornado Wind Speeds From Tree Damage Patterns. (Under the direction of Dr. Allen J. Riordan)

Near surface wind speeds are important in the understanding of a tornado's structure, but can be difficult to measure directly. However, it is possible to get an estimate of near surface tornado wind speeds by examining tree damage patterns. Johannes Letzmann (1923) hypothesized that different combinations of tangential, radial and forward velocities yield different damage patterns.

The F-scale classifies tornadoes based on the damage they cause to buildings. Therefore when a tornado passes through a forest, it cannot be assigned an F-number.

A model (TornTree) has been developed that creates a vortex based on the combined Rankine vortex and moves it over a forest. The trees are modeled using a modified version of the HWIND model from Peltola et al. (1999). The model is run on Scots pine to validate it with the HWIND model. Due to their prevalence in the Southeast United States, the model is also run on loblolly pine. The model calculates a turning moment for each tree based on the vortex wind speed at that instant and compares it to the resistance to stem breakage for that tree. If the turning moment is greater, then the tree's stem is broken and the tree falls.

The model outputs graphical files of the vortex winds and the damage pattern. This hypothetical damage pattern can be compared with actual damage patterns to estimate the near surface wind speed of the tornado and to estimate its F-number.

DEVELOPING A MODEL TO ESTIMATE TORNADO WIND SPEEDS FROM TREE DAMAGE PATTERNS

by

ANDREW P. HOLLAND

A Thesis submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Master of Science

DEPARTMENT OF MARINE, EARTH AND ATMOSPHERIC SCIENCES

Raleigh, NC
2000

APPROVED BY:

Carlyle Franklin

Jerry Davis

Allen J. Riordan
Chairman of Advisory Committee

DEDICATION

To the loves of my life,

My wife Karen---

Thank you for your love and support through this.
Without you this would not have been possible.

My daughter Emma—

I hope one day you can forgive me of the time this
thesis took away from our time together,
and realize that I did it so that you would have
a better life.

I love you both.

BIOGRAPHY

Andrew (Andy) P. Holland was born in Raleigh, North Carolina on 20 September 1975. Soon after birth his family moved to Hallsboro, North Carolina near Wilmington. Being near the coast, Andy experienced quite a few thunderstorms in the summer. Andy's mother would have to peel him away from the windows where he liked to sit and watch the storms. Andy's love of weather was broadened when five years later he moved just south of Charlottesville in the mountains of Virginia. There he got a taste of cold winters and lots of snow.

In 1988, Andy returned to North Carolina to live in the small town of Wendell. He enrolled at North Carolina State University in 1993 to study Meteorology. Andy was very active in the Marching Band where he met his lovely wife, Karen. Four years later he was accepted into graduate school at North Carolina State University to pursue a Master of Science degree in Atmospheric Sciences. Prior to completion of his degree Andy was hired by the Environmental Programs Group at MCNC in the Research Triangle Park, North Carolina, as an Air Quality Meteorologist.

Andy was married in May 1998 to the former Karen Brooks. They are the proud parents of their 9-month old daughter, Emma Maie.

ACKNOWLEDGEMENTS

First, to my wife Karen, you have been a pillar of strength and support for me during my graduate school experience. Thank you for your patience, since this experience was a little longer than we both expected. Your support and opinions have been very helpful in the development of this thesis.

To my daughter Emma, thank you also for your help with this thesis. I will cherish the times I held you while I sat at the computer typing away, and erasing all of your additions.

Thanks also to my parents, who have always encouraged me to keep on learning. You have both been and will continue to be a strong influence in my life.

To Dr. Al Riordan, my advisor for seven years and committee chair, thank you for all of your help and for bringing this project to my attention. I have enjoyed this project and working with you so much that it does not matter to me that it was not funded.

I would also like to thank my other committee members, Dr. Carlyle Franklin and Dr. Jerry Davis. Thank you for all of your advice and comments that helped shape this thesis. Also, an extra thank you goes to Dr. Franklin for helping me understand some basic forestry principles.

I would also like to acknowledge Dr. Heli Peltola from the University of Joensuu in Finland for letting me incorporate her tree model into my model.

Thank you also for your help in understanding your model, for the papers you sent me and for promptly answering my many e-mails.

I would also like to thank Ed Bilicki, supervisor of the Environmental Programs Group at MCNC, for hiring me before I completed graduate school and for allowing me some time to complete my thesis.

TABLE OF CONTENTS

APPENDIX B – Model Walk Through

APPENDIX C – Sample Files From the Model

APPENDIX D – Model Program

LIST OF TABLES

Page

LIST OF FIGURES

Page

LIST OF ABBREVEATIONS AND SYMBOLS

TornTree	model created in this thesis
HWIND	model to determine critical wind speeds (Peltola et al. 1999)
h	tree height in meters
dbh (DBH)	diameter at breast height in meters
U_{upr}	wind speed to uproot in m/s
U_{break}	wind speed to cause stem breakage in m/s
x, y	coordinates of grid points
r	distance from origin to each grid point (radius)
V_{tan}	tangential velocity
V_{max}	user's inputted maximum tangential velocity
R_{max}	radius of maximum tangential velocity
V_r	radial velocity
V_{rmax}	user's inputted maximum radial velocity
u(z)	wind speed at height z in the canopy in m/s
u(m)	wind speed at one tree height above the canopy in m/s
z	height above the ground in meters
z_0	roughness length in meters
Fw(z)	force due to the wind in N
Cd	drag coefficient
?	density of air in kg/m^3
A(z)	area of one-meter segment of tree in m^2
S_t	streamlining function
Fg(z)	force due to the mass of the tree in N
M(z)	mass of one-meter segment of tree in kg
g	gravitational acceleration
$T_{max}(z)$	turning moment for one-meter segment in Nm
$Gust_{factor}$	relates the mean wind loading to the extreme values

Gap_{factor}	takes into account the effect of the size of the upwind gap on the magnitude of the turning moment
$x(z)$	horizontal displacement of the stem in meters
$l(z)$	distance from treetop to the middle of tree segment in meters
MOE	modulus of elasticity in Pa
MOR	modulus of rupture in Pa
I	moment of inertia in m^4
$Gust_{mean}$	mean wind loading at stand edge
$Gust_{max}$	extreme gust values
Gap_{mean}	mean wind loading for infinite upwind gap
Gap_{max}	extreme wind loading values
s	tree spacing in meters
x	distance from stand edge in tree heights
gap_{size}	size of the upwind gap in tree heights
T_{max}	total turning moment at stem base
$STEM_{res}$	resistance to stem breakage
G_{max}	ratio of rotational to translational components of motion
?	angle measured counterclockwise from a vector pointing toward the center of the vortex to the relative wind vector at that point

1. Introduction

1.1 Overview

Tornadoes are arguably one of nature's most destructive forces. They were responsible for an average of 91 fatalities per year and over $42,000,000 in total property damage in the U.S. from 1950 to 1994 (NOAA Storm Prediction Center [SPC] 1995). During this time, meteorologists have been trying to better understand the structure of tornadoes. One way to do this is to measure their maximum winds near the surface, a very difficult endeavor because winds from tornadoes destroy most direct measurement devices. To combat the problem, research has been done to estimate tornado wind speeds near the surface from Doppler radar analysis (Zrnic et al. 1985).

Another, and much less expensive and time-consuming, way to estimate tornado maximum wind speeds is to study the damage they produce. The F scale, proposed and developed by T. Fujita (1981), is a scheme for classification of tornadic wind speed based on the degree of damage that it causes to buildings. By estimating damage severity, one can obtain the approximate wind speed of the tornado and its direction of motion.

The problem with this method is that it can only be used where there are damaged buildings. Thus, there is no simple wind classification scheme that can be used in open areas and those of natural vegetation.

However, with knowledge of the types of trees in a forest one can determine the amount of wind the trees can withstand, and because of the density of targets, one can obtain a detailed damage pattern.

Letzmann (1923, also reviewed by Peterson 1992) suggested that when a tornado moves through a forest it produces a damage pattern that is related to both the wind speed and forward speed of the tornado. By creating a simple, analytical model of a vortex, Letzmann was able to simulate tree damage patterns and compare them to actual damage patterns and therefore estimate the wind speed and forward speed of a tornado.

1.2 Objectives

The goals of this project are to develop a simple objective method to aid in estimating maximum, near-surface tornado wind speeds from damage patterns in forested areas and to produce analyses of tree-fall patterns that can be easily generated by the National Weather Service and damage assessment teams. In support of the goal, the main objective is to create a model that will produce a diagram of the tree-fall pattern when given a number of simple variables such as the maximum rotational wind speed, the maximum inflow (radial) speed, the forward speed, and the radius of the tornado. This model will combine a vortex model based on the combined Rankine vortex and a modified tree model based on the HWIND model designed by Peltola et al. (1999).

The model will be run on Scots pine trees using the data from Peltola et al. (1999) in order to validate it. The model will then be run using loblolly pine data based on their prevalence in the southeastern United States (Hedden et al. 1995).

1.3 Literature Review

1.3.1 Vortex models

There are several classes of vortex models used to describe tornadoes; physical, numerically simulated and analytical. Ward (1972) created a physical model of a tornado with laboratory apparatus that used fans to create an adjustable airflow that simulated a small-scale tornado and allowed Ward to measure the inflow angle compared to the tangential airflow. The drawback of this model was that it did not allow for measurement of the wind speeds.

Turner (1966) also created a physical model of a tornado in the laboratory. A stream of gas bubbles was released into a rotating tank of fluid. The gas bubbles created an updraft and consequently circulation. From this laboratory simulation Turner was able to produce profiles of tangential velocity (Figure 1.3.1).

Numerical models are designed to simulate tornadoes by integrating the equations of motion and imposing boundary conditions.

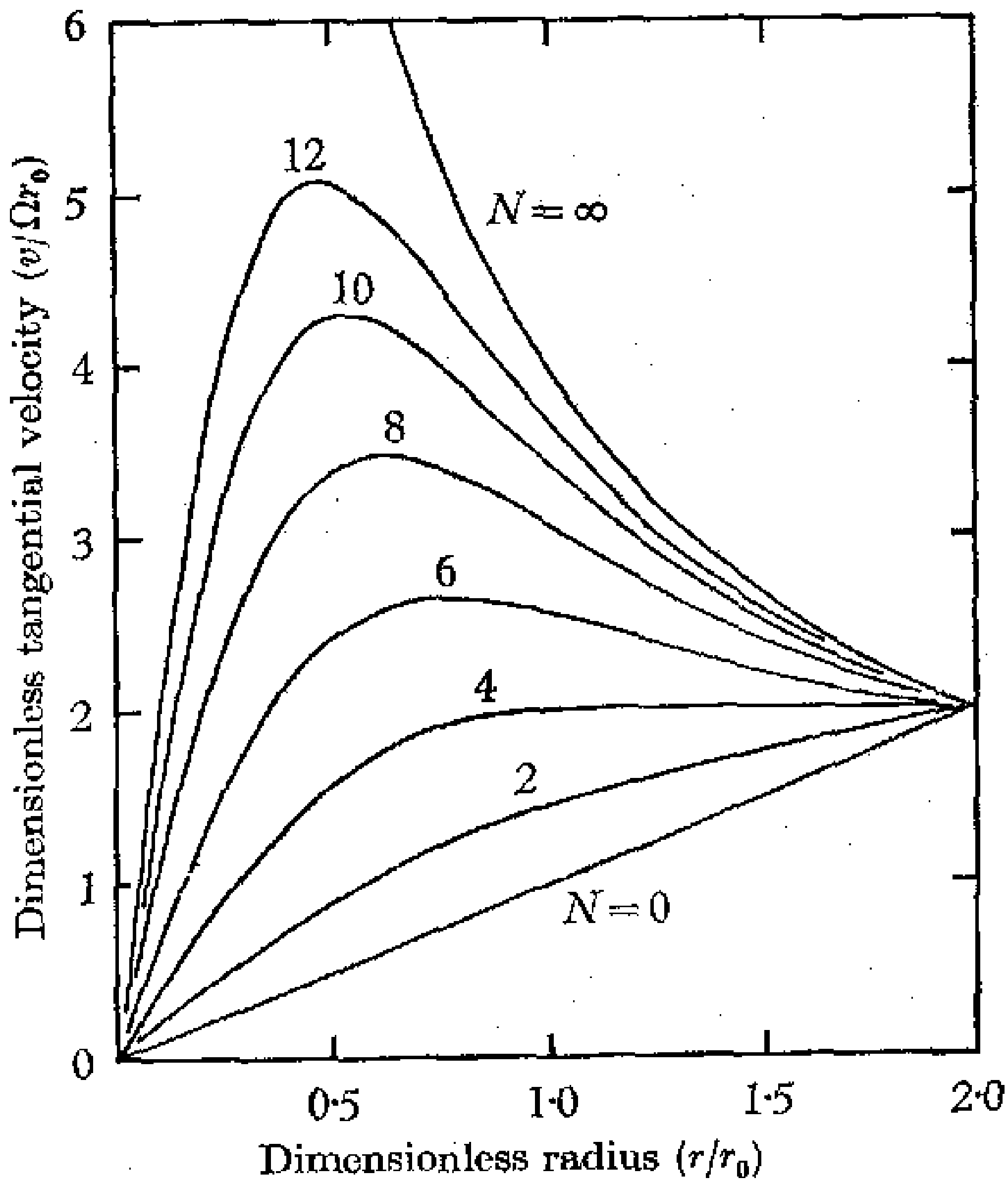


Figure 1.3.1: Tangential velocity profiles from Turner's laboratory vortex with varying values of Reynolds number (from Turner 1966).

Lewellen (1993) and Trapp and Fiedler (1993) created numerical simulations of tornadoes and found that they correspond well to the physical models, but require a large amount of code and are time-consuming to run.

The simplest analytical vortex model is the combined Rankine vortex (Lewellen 1993). In this type of model, the radial and vertical velocities are set equal to zero, leaving a two-dimensional, tangential wind field. The vortex contains an inner region of solid-body rotation and an outer region of potential flow. In the inner region, tangential velocity increases linearly with radius to a maximum, while in the outer region it decreases inversely with radius (Figure 1.3.2). The Rankine combined vortex was developed in 1882 (Rankine 1882), but has been used recently to develop algorithms of mesocyclone and tornado detection for Doppler radar (Vandersip 1998).

Letzmann (Peterson 1992) studied several aspects of the tornado, beginning with a tornado climatology of the Baltic States and ending with constructing a model for a translating vortex. Letzmann used a combined Rankine vortex to estimate the tangential flow of a tornado. He then added the radial wind flow pattern, assuming that it changed with radius in a similar fashion as that of the tangential velocity. By adding a translational wind field to the tangential and radial velocities Letzmann produced the resultant wind speed and direction for every point in and around the vortex. Letzmann determined two variables, G and ?, which

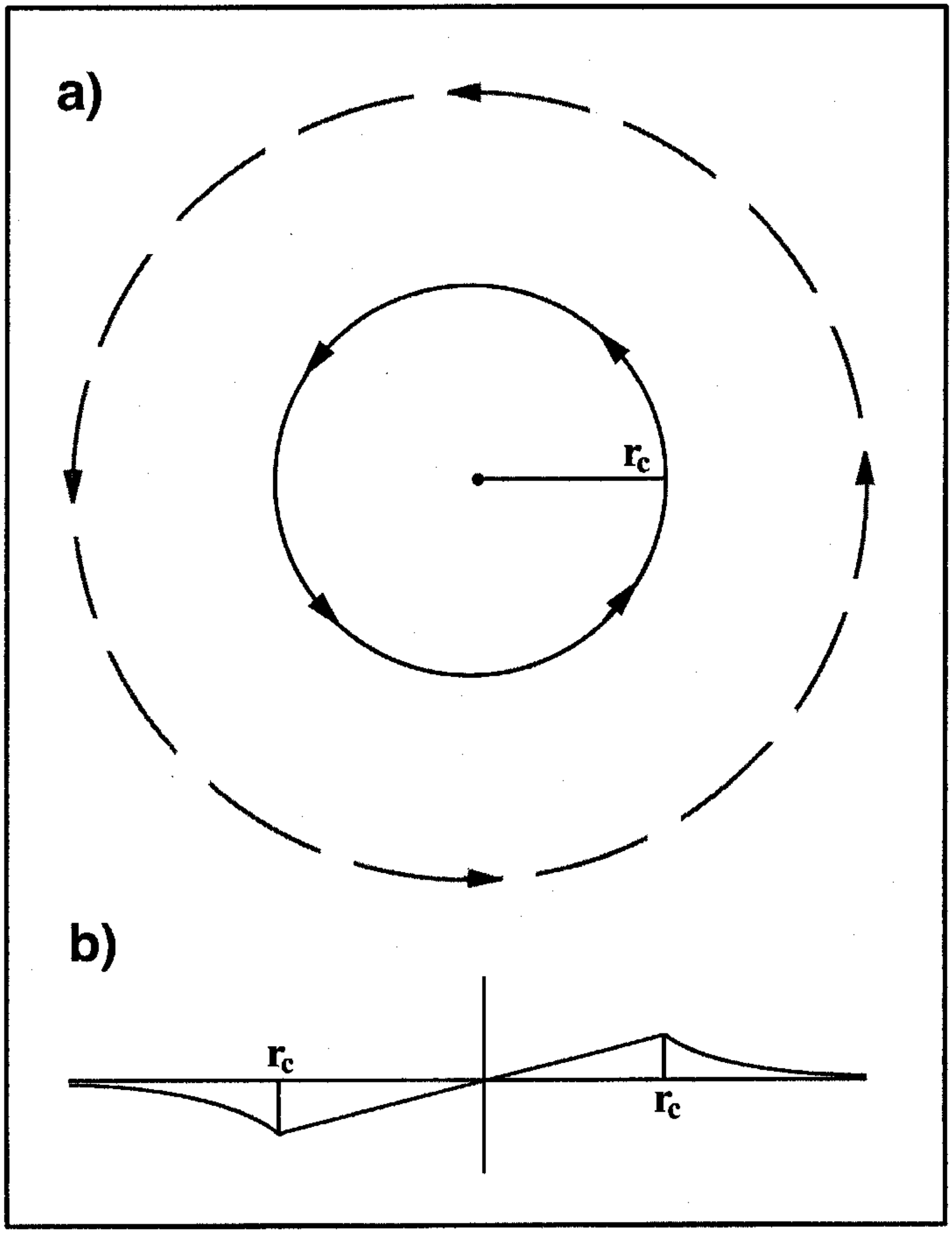


Figure 1.3.2: Schematic of (a) an axisymmetric vortex and (b) its combined Rankine velocity distribution. The radius of maximum velocity is denoted as r_c (from Vandersip 1998).

he could vary to create different wind field patterns. The variable G is the ratio of rotational to translational components of motion and ? is the angle measured counterclockwise from a vector pointing toward the center of the vortex to the relative wind vector at that point. By varying G and ?, Letzmann, by hand, calculated many different wind field patterns.

Letzmann hypothesized that by examining tornado damage patterns, he could deduce the vortex structure. He created hypothetical damage patterns based on his vortex model. He assumed that a tree would fall at some critical velocity in the instantaneous direction of flow and that there was no interaction among the trees. He used his model to compare his hypothetical damage patterns with actual damage and estimated the tangential and radial wind speeds and the forward speed of the tornado. Some generalized swath patterns for different combinations of rotational and translational speeds are shown in Figure 1.3.3.

Roth (1990) created a vortex model to study the damage patterns from the Raleigh, NC tornado that occurred on 28 November 1988. His vortex model was very similar to Letzmann's. Roth combined a Rankine vortex with an inflow field and translational motion. He then moved this model vortex over a field of uniformly spaced, fixed targets. These targets represented trees and were assigned a value of damage on a random basis. These values were based on failure thresholds determined by Fujita (1978) for trees typical to the Raleigh area. If the wind speed exceeded this failure threshold, the tree was considered to be downed.

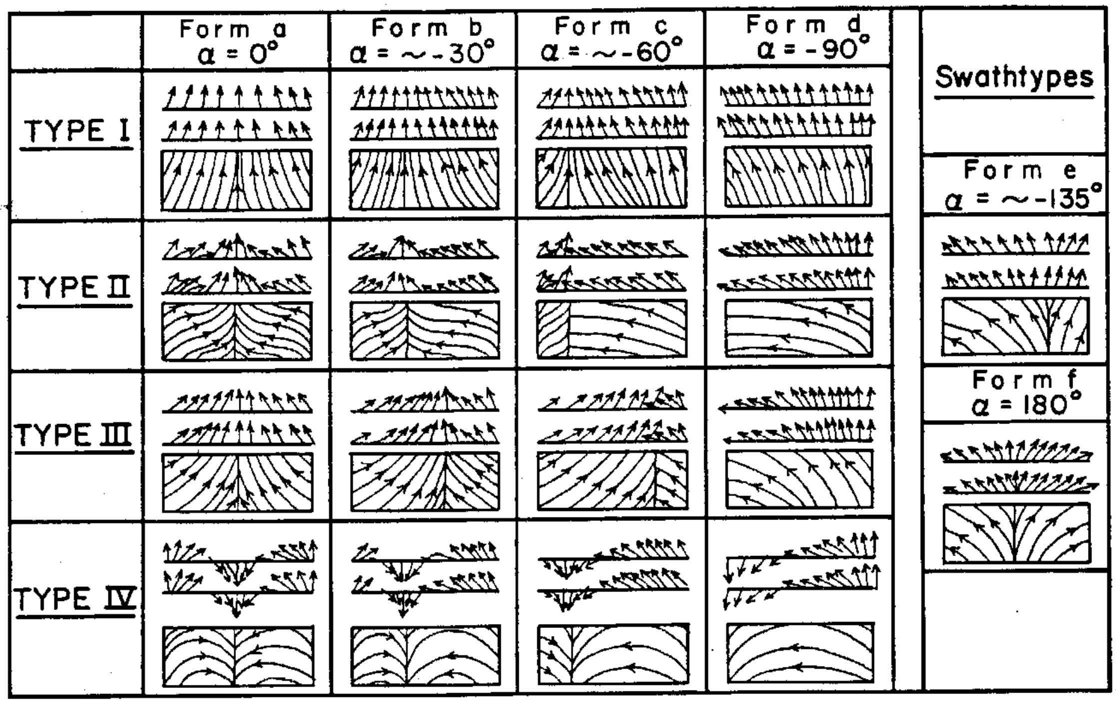


Figure 1.3.3: Generalized swath patterns for different vortex combinations of inflow and ratios of rotational to translational components of motion. G increases from Type I to Type IV. ? =0? is purely radial flow and ? =-90? is purely tangential flow (from Peterson 1992).

The model produced a graphical file showing the orientation of all downed trees, which Roth then compared to actual damage patterns from the Raleigh tornado. Roth concluded that his model did an adequate job of simulating the damage patterns associated with the Raleigh tornado.

There are, of course, limitations to Letzmann's and Roth's models. Letzmann's model did not include any information about trees. He assumed that the trees fell at some specified critical velocity (Peterson 1992). Roth's model improved on this by assigning a critical velocity to each tree based on trees prevalent to the Raleigh, NC area. The new model, TornTree, described in this thesis, will improve on this further. The TornTree model will include a detailed tree model that will determine the critical velocity for each tree in the forest. The model can be used for many tree species and locations.

1.3.2 Recent advances in forestry

Much of the research done on the response of trees to the wind has been done in Europe. Wood (1995) used wind tunnel experiments on model trees and pulling experiments in the field to calculate the maximum stresses that different species of trees can withstand. Mattheck and Breloer (1994) developed a simple windthrow model that calculates the additive wind and gravity forces acting upon the tree. If the sum of these forces exceeds the tree's resistance to uprooting, the tree will fall.

Some research on the response of trees to wind has been done in the United States. Researchers at Clemson University have used field studies to calculate the bending moments of loblolly pine (*Pinus taeda* L.) trees (Fredericksen et al. 1993; Hedden et al. 1995). In these studies, loblolly pine trees were toppled using a winch and cable system. The force need to break the stem, the critical turning moment, was measured for trees of different sizes and ages.

Peltola et al. (1997, 1999), Peltola and Kellomäki (1993) and Peltola (1996) developed a mechanistic tree model that predicts the critical turning moment and wind speed that will cause Scots pine (*Pinus sylvestris* L.), Norway spruce (*Picea abies* (L.) Karst.) and birch (*Betula* spp.) to be uprooted or cause the stem to break. In their model, the horizontal force due to the wind and the vertical force due to gravity are calculated for each one-meter segment on the stem and crown. The maximum turning moment is determined for the tree from the above forces, but must be adjusted for the gustiness of the wind to provide a value for the extreme wind loading for a particular wind speed.

In this model, each tree has a resistance to uprooting and stem breakage. The former is derived from the weight of the root-soil plate (Peltola and Kellomäki 1993). If the maximum turning moment is greater than this resistance, the tree is uprooted. The resistance to stem breakage is based on the assumption that the stress due to the wind on the outer fibers of a tree is constant between the base of the canopy and the butt of the tree. The resistance is calculated from the diameter at breast height and the modulus of rupture (Jones 1983; Morgan and Cannell 1994). If the maximum turning moment is greater than this resistance the tree's stem is broken.

Peltola et al. (1997, 1999) calculated the critical wind speeds for Scots pine, Norway spruce and birch. The model begins with a wind speed of 1 ms^{-1}, calculates the turning moment and compares it to the resistances. If the turning moment is less, 0.1 ms^{-1} is added to the wind speed and the calculations are repeated until the resistance is exceeded. Calculations of maximum turning moments from this model were in good agreement with the critical turning moments calculated from tree pulling experiments (Peltola et al. 1999). Table 1 gives the wind speeds needed to break stems and uproot Scots pine at different distances from the stand edge assuming an infinite upwind gap. These wind speeds are not the extreme wind values due to gustiness; they are the mean wind values above the stand, which are approximately 85% of the gusting speed (Peltola et al. 1993).

Table 1.3.1. Critical wind speeds at different distances (in tree heights, *h*) from the edge of a Scots pine stand 20m in height, 0.20m diameter at breast height and 3.8m spacing (Peltola et al. 1999).

Parameter	0h	1h	2h	3h	4h	5h
U_{upr} (ms^{-1})	18.0	27.0	31.0	32.0	32.0	32.0
U_{break} (ms^{-1})	22.0	30.0	34.0	35.0	36.0	36.0

2. Model description

2.1 General description

The purpose of this model is to simulate tree damage patterns caused by a tornado moving through a forest. To do this, a model vortex is created and then moved over a model forest. The vortex model is based on the combined Rankine vortex created by combining a rotating, radial and translating wind field. As a first step in model development, the vortex is two-dimensional and steady state, the terrain is considered to be flat and there is no interaction among the trees.

The tree model is based on the mechanistic tree model designed by Peltola et al. (1999). A resistance to stem breakage is calculated for each tree. A resistance to uprooting is not calculated since the TornTree model will only be run for pine trees, which are more likely to suffer stem breakage (Fredericksen et al. 1993; Hedden et al. 1995). A turning moment is then calculated based on the wind speed and tree characteristics. If this turning moment is greater than the resistance, the tree's stem is broken. A graphical output file is created showing the orientation of the fallen trees.

The user inputs an initial guess of the maximum tangential velocity, the maximum radial velocity, the forward speed and the radius of the tornado. The model will output a graphical file of the orientation of the fallen trees.

2.2 Vortex model description

The model begins with a combined Rankine vortex. Each point in the grid is assigned x,y coordinates. These coordinates can be transformed to polar coordinates, where r is the distance from the origin to that point, i.e.:

$$r \; ? \; \sqrt{x^2 \; ? \; y^2} \, . \tag{1}$$

From this, the tangential velocity can be calculated at each point using:

$$V_{\tan} \; ? \; (V_{\max} r)/R_{\max} \text{, when } r \; ? \; R_{\max}, \tag{2}$$

$$V_{\tan} \; ? \; (V_{\max} R_{\max})/r \text{, when } r \; ? \; R_{\max}, \tag{3}$$

where R_{max} is the radius of the maximum tangential velocity. Figure 2.2.1 shows the model output of a combined Rankine vortex.

Letzmann (Peterson 1992) and others, (Ward 1972; Zrnic et al. 1985, Turner 1966) suggest that the radial velocity varies with radius in a similar fashion, that is:

$$V_r \ ? \ (V_{r\max} r)/R_{\max}\ , \text{ when r ? } R_{max}, \tag{4}$$

$$V_r \ ? \ (V_{r\max} R_{\max})/r\ , \text{ when r ? } R_{max}. \tag{5}$$

Adding the tangential and radial velocities produces Figure 2.2.2 to represent the mean wind field for a stationary tornado. Finally, to make the model simulate a moving tornado, the forward motion is added to the wind field (Figure 2.2.3). The wind speed at each grid point is then passed on to the tree model.

2.3 Tree model description

The tree model used in the TornTree model is based on Peltola et al. (1999). First, the model extrapolates the two-dimensional wind field from above into a three-dimensional field in the canopy, assuming a log profile of the form (Figure 2.3.1):

$$u(z) \ ? \ u(m) \ ? \ \frac{\ln(\frac{z}{z_0})}{\ln(\frac{h}{z_0})}, \tag{6}$$

where u(z) is the wind speed at height z, u(m) is the input wind speed from the model which is assumed to be one tree height above the canopy and z_0 is the roughness length, which is taken to be 0.06 times

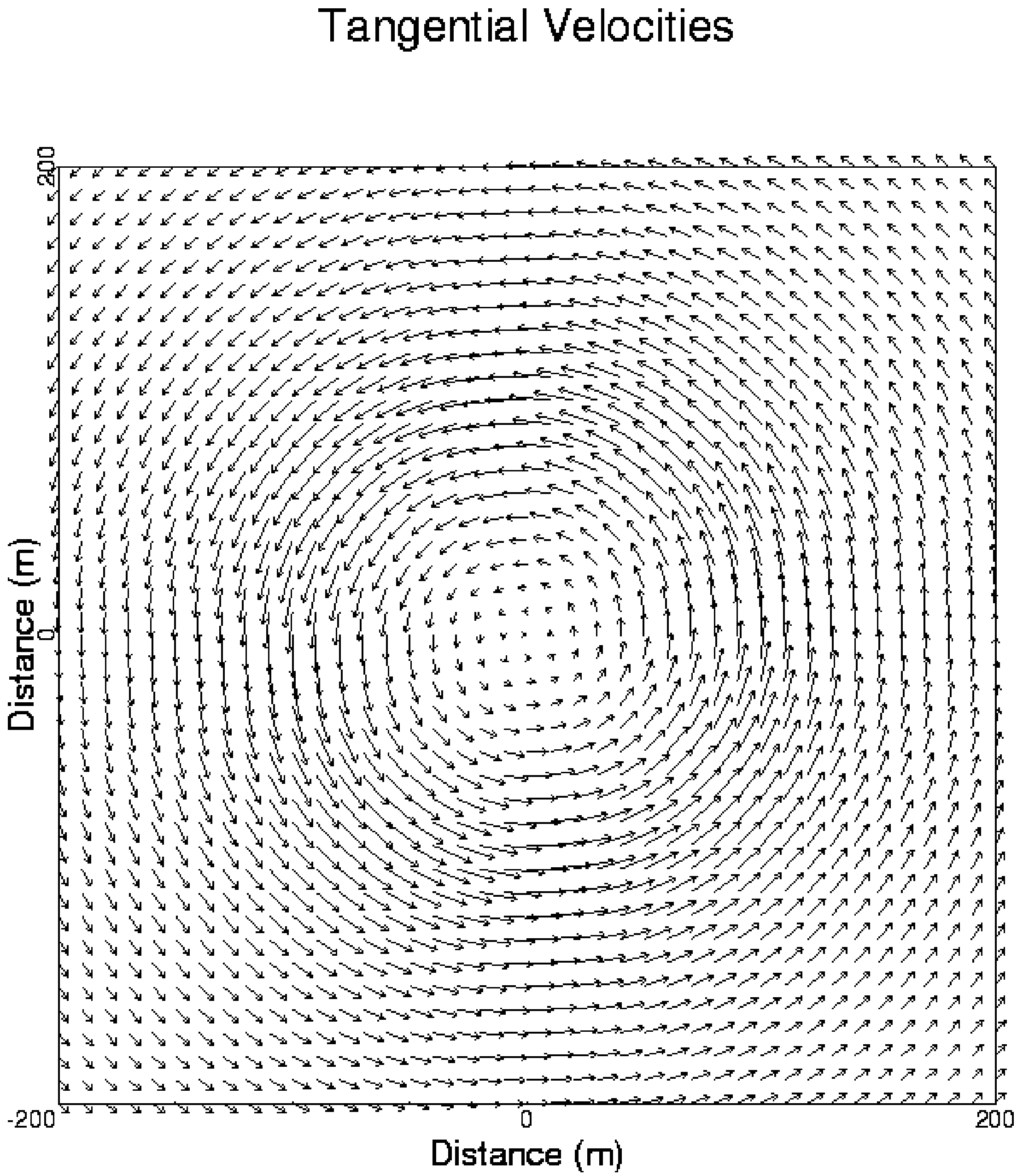


Figure 2.2.1: Model output of tangential velocities, V_{max} = 50 ms^{-1}.

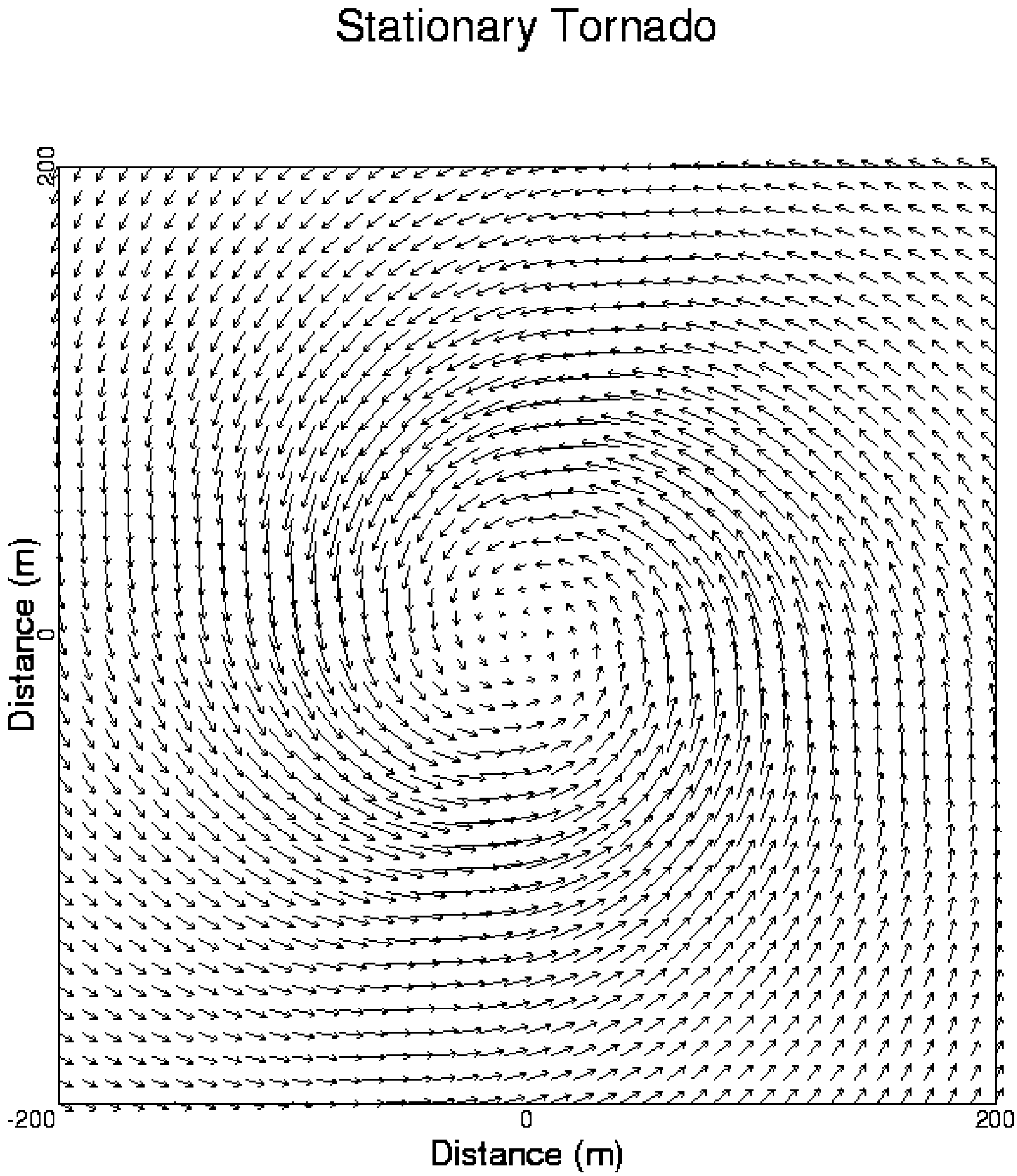


Figure 2.2.2: Figure 2.2.1 plus radial velocities.

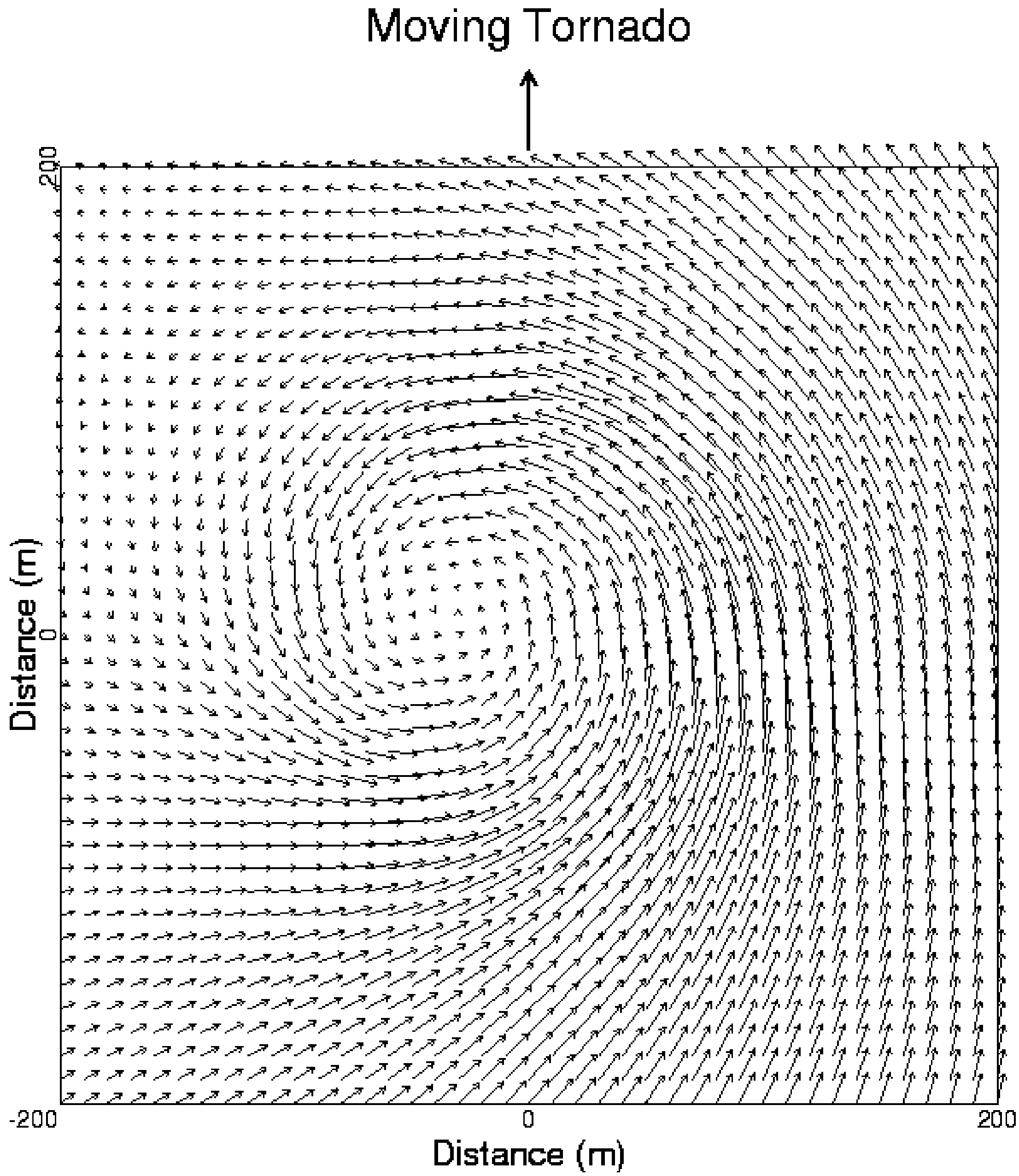


Figure 2.2.3: Figure 2.2.2 plus forward motion, V_{forw} = 15 ms^{-1}.

the tree height (Peltola et al. 1999). The tree is divided into one-meter segments and through (6) the wind speed is calculated at each height, z.

Second, the model calculates the forces affecting the tree. The first force considered is the force due to the wind. The mean wind force, Fw (N), for each one-meter segment is given as (Monteith 1975),

$$Fw(z) \, ? \, \frac{C_d \, ? \, ? \, ? \, u(z)^2 \, ? \, A(z)}{2} \qquad (7)$$

where C_d is the drag coefficient for the tree, ? is the density of air and A(z) is the projected area of the tree against the wind.

To calculate the projected area of the crown against the wind, Peltola et al. (1999) assumed a crown shape of two triangles for Scots pine and birch, and one triangle for Norway spruce. The bases of the triangles are equal to twice the length of the longest branch in the crown (Hakkila 1971; Peltola and Kellomäki 1993). For loblolly pine, the crown is approximated as two triangles also, but the upper triangle is larger than the lower, to differentiate between the living and dead crown (Franklin 2000). As the wind speed increases the canopy streamlines. Peltola et al. (1999) determined that the reduction in crown area is 20% when the wind speed is less than or equal to 11 ms^{-1} and 60% for wind speeds greater than 20 ms^{-2}. Peltola and Kellomäki (1993) give the following equation to determine the streamlining factor, S_t, between these wind speeds as (u(z) is from equation 6):

$$S_t \, ? \, \frac{10}{u(z)} \, ? \, 0.10 \qquad (8)$$

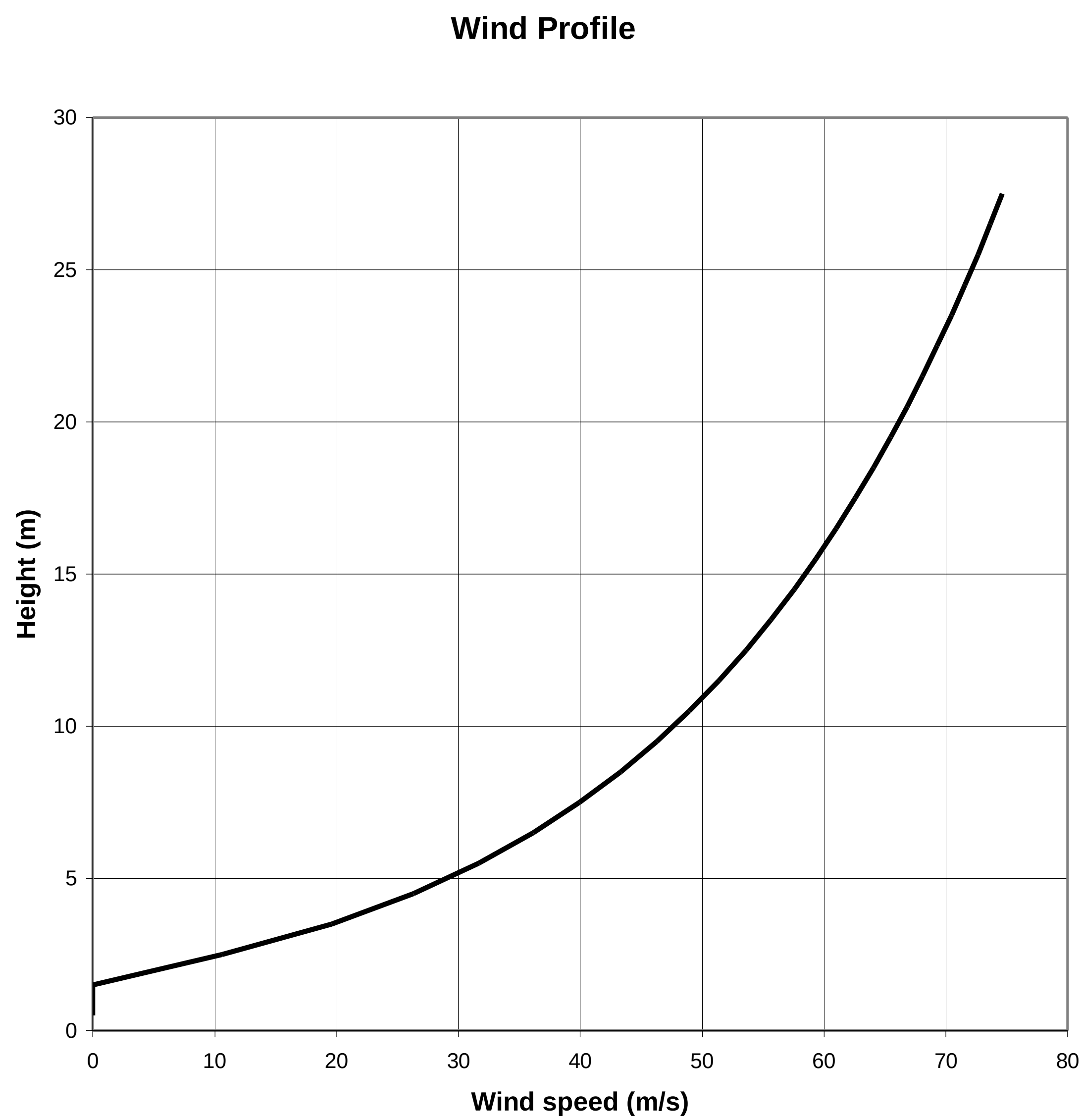


Figure 2.3.1: Wind profile in the canopy.

Since the force due to the wind bends the tree, a force due to gravity becomes important. This force is also calculated for each one-meter segment of the tree by,

$$Fg(z) ? M(z) ? g \qquad (9)$$

where g is the acceleration of gravity and M(z) is the green mass of the stem and crown (Grace 1977; Jones 1983; Petty and Swain 1985; Peltola and Kellomäki 1993).

The mass of each stem segment is calculated from the green density of the stem and the volume of the segment (Clark 1991; Laasasenaho 1982; Peltola and Kellomäki 1993). The vertical distribution of crown mass is assumed to be similar to that of the crown area projected against the wind and is proportional to the stem mass (Baldwin 1987; Peltola and Kellomäki 1993). See Appendix A for a detailed description of the area and mass distribution calculations.

The stem base turning moment can be calculated for each segment from the stem and crown mass, the wind force and the horizontal deflection of the stem (Figure 2.3.1), but it must be adjusted by factoring in the gustiness of the wind and the distance between the trees. The maximum turning moment is then,

$$T_{\max}(z) ? Gust_{factor} ? Gap_{factor} ? \{Fw(z) ? z ? Fg(z) ? x(z)\}, \qquad (10)$$

where z is the height of the segment and x(z) is the horizontal displacement of the stem, which is assumed to be directly proportional to

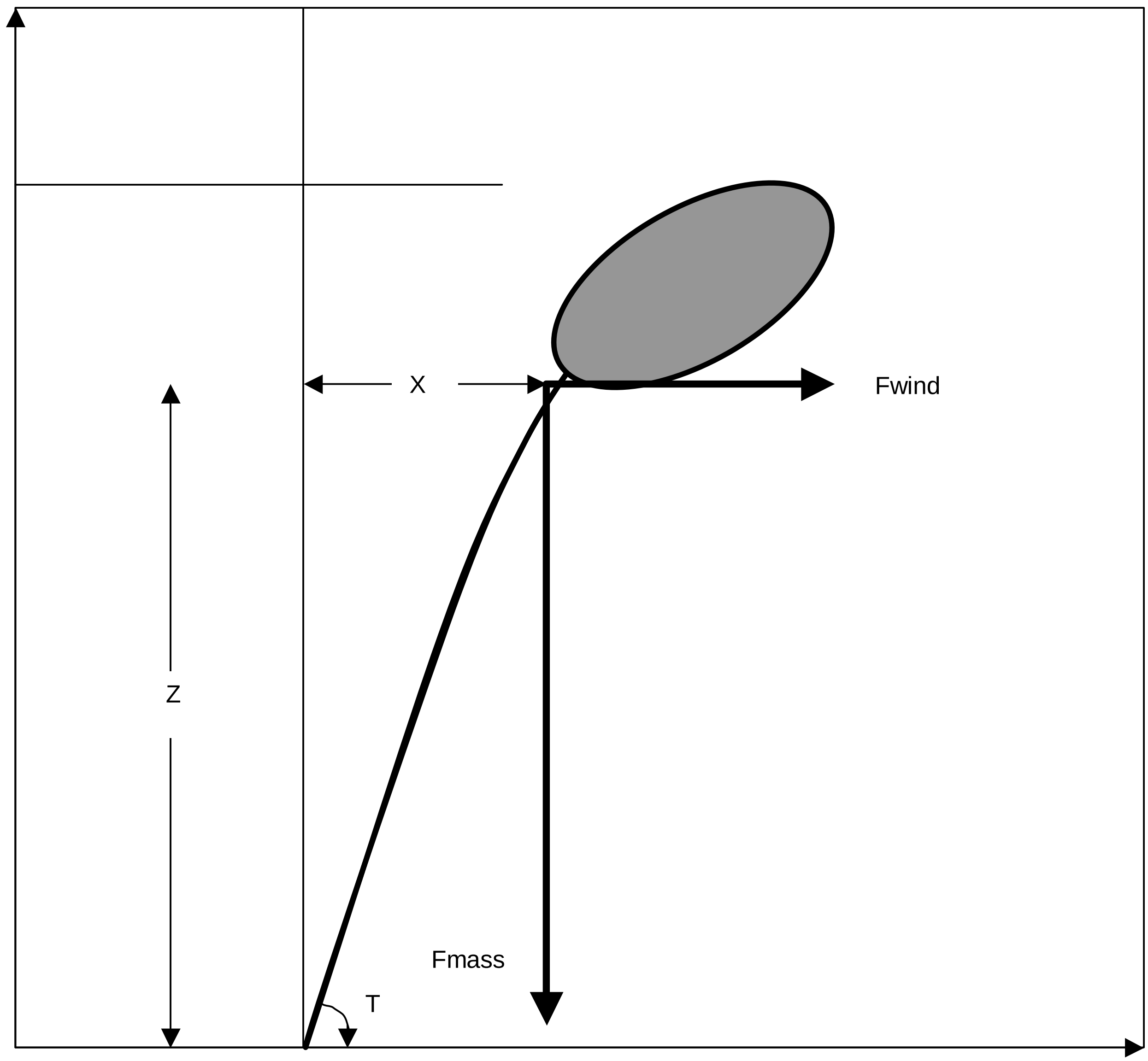


Figure 2.3.2: Wind force (Fwind) and gravity (Fmass) acting upon a tree cause the total turning moment (T) at the stem base (after Peltola et al. 1997).

the wind force and inversely proportional to the stem's stiffness (Pennala 1980). This is given by,

$$x(z) ? \frac{Fw(z) ? a^2 ? h ? (3 ? a/h ? 3 ? l(z)/h)}{6 ? MOE ? I}, \text{ when z ? a} \qquad (11)$$

$$x(z) ? \frac{Fw(z) ? a^3 ? (2 ? 3 ? (l(z) ? b)/a ? (l(z) ? b^3)/a^3)}{6 ? MOE ? I}, \qquad (12)$$

when z ? a,

where l(z) is the distance from the tree top to the middle of each one-meter segment, MOE is the modulus of elasticity (Pa), I is the area moment of inertia of the stem (m^4), a is the distance from ground level to crown center and b is the distance from crown center to the tree top.

The $Gust_{factor}$ is determined empirically from wind tunnel experiments and is dependent on the tree height, tree spacing and distance from the stand edge. It represents the changes in mean wind loading to the extreme normalized values (Gardiner and Stacey 1995; Gardiner et al. 1997):

$$Gust_{mean} ? [0.68(s/h) ? 0.0385] ? [?0.68(s/h) ? 0.4785][1.7239(s/h) ? 0.0316]^{x/h} \qquad (14)$$

$$Gust_{\max} ? [2.7193(s/h) ? 0.061] ? [?1.273(s/h) ? 0.9701] ? [1.1127(s/h) ? 0.0311]^{x/h} \qquad (15)$$

$$Gust_{factor} ? {Gust_{\max}}/{Gust_{mean}(x ? 0)} \qquad (16)$$

where s is the tree spacing and x is the distance from the forest edge in tree heights (i.e., x=0h for stand edge and x=1h for one tree height into the stand). Figure 2.3.2 shows the effect distance from stand edge has on the turning moment. This relationship assumes an infinite upwind open area. Therefore, a gap factor must be determined to take into account the effect of different upwind gap sizes on the wind loading (Gardiner and Stacey 1995; Gardiner et al. 1997).

$$Gap_{mean} = \frac{0.001 + 0.001(gap_{size})^{0.562}}{0.00465} \qquad (17)$$

$$Gap_{\max} = \frac{0.0072 + 0.0064(gap_{size})^{0.3467}}{0.0214} \qquad (18)$$

$$Gap_{factor} = Gap_{\max} \big/ Gap_{mean}(gap_{size} = 10) \qquad (19)$$

where gap_{size} is the size of the upwind gap in tree heights. The effect of the upwind gap can be seen in Figure 2.3.3.

The maximum turning moment at stem base can be calculated by summing the turning moments at each one-meter segment over the height of the tree:

$$T_{\max} = \sum_{z=0}^{h} T_{\max}(z) \qquad (20)$$

This total turning moment is compared with the resistance to stem breakage for each tree.

The resistance to stem breakage is based on the modulus of rupture for the tree. It is calculated at breast height (z=1.3m) and is given by,

$$STEM_{res} = \frac{\pi}{32} \times MOR \times DBH^{3} \qquad (21)$$

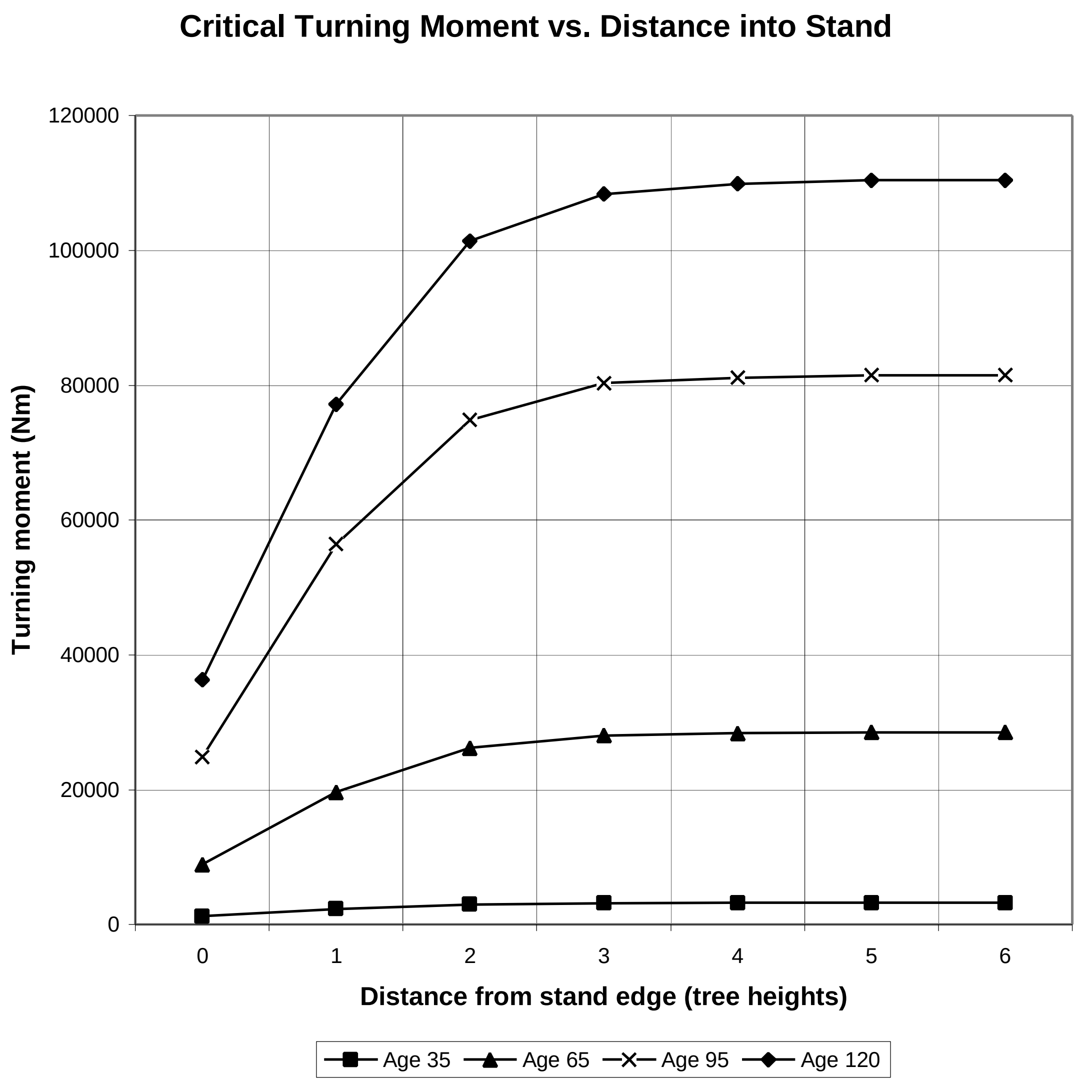


Figure 2.3.3: Critical turning moments for stem breakage of Scots pine with distance from stand edge.

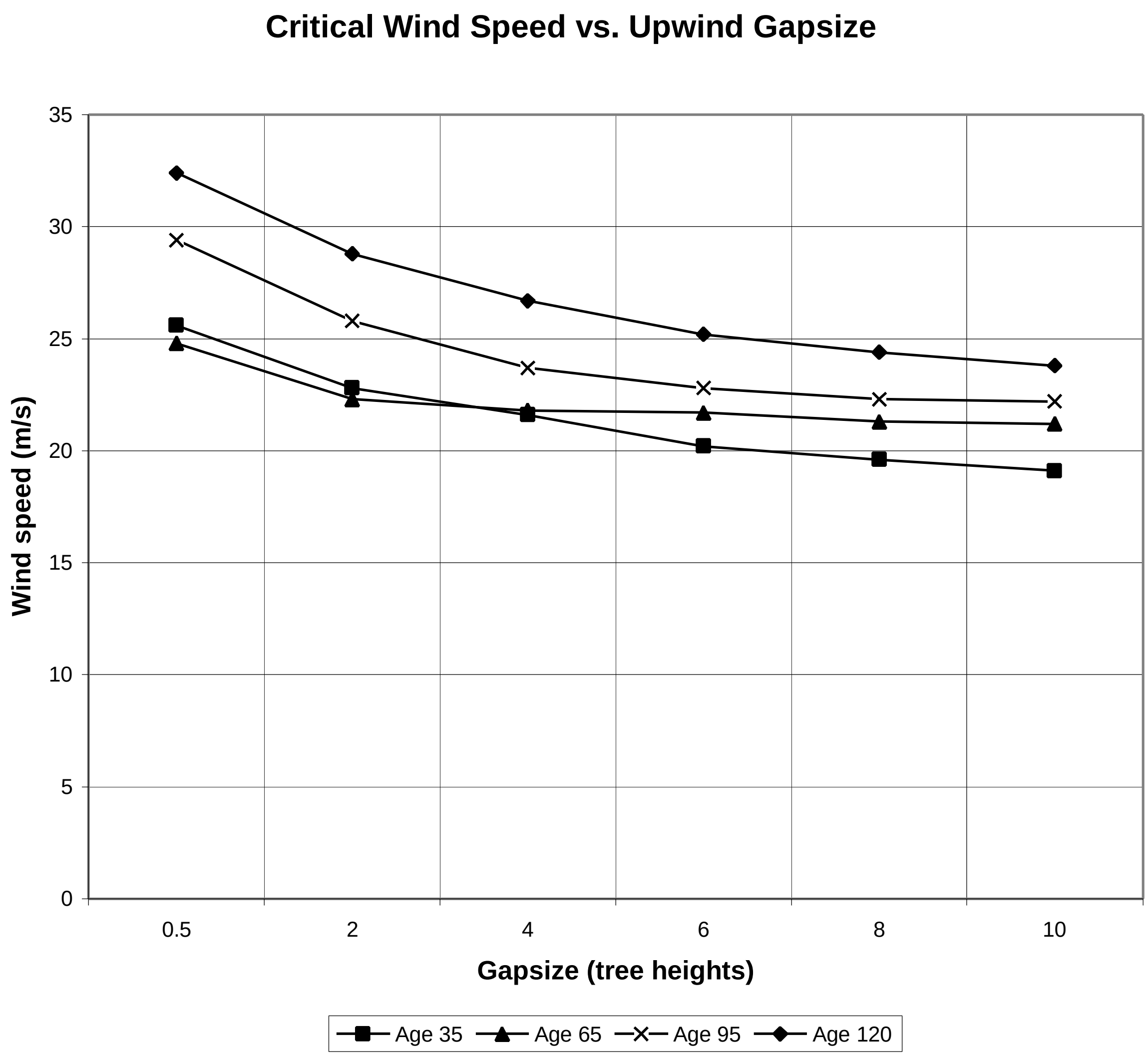


Figure 2.3.4: Critical wind speed for stem breakage of Scots pine with different upwind gap sizes at stand edge.

where MOR is the modulus of rupture (Pa) and DBH is the diameter at breast height (m) (Jones 1983; Morgan and Cannell 1994).

The model moves the tornado vortex over the forest calculating the turning moment for each grid point and comparing it to the resistance to stem breakage. If the turning moment exceeds the resistance, the tree is considered to be downed. The model outputs a graphical file showing the orientation of the downed trees. By varying the input variables of the model, many different damage patterns can be obtained. These patterns can then be compared with photographs of actual tornado damage to estimate the wind speed and forward speed of the tornado.

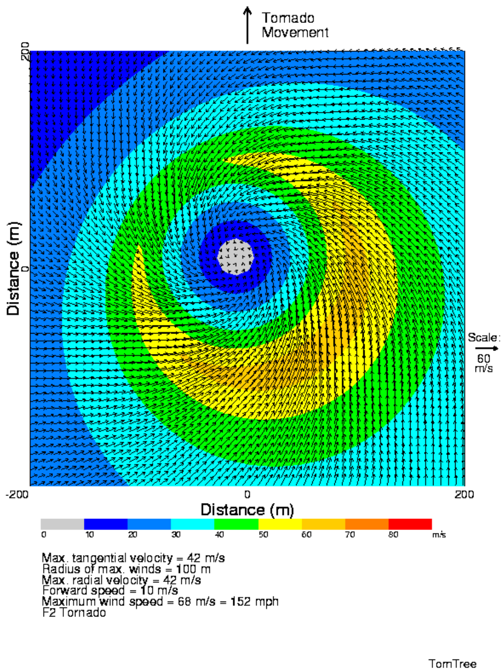
3. Results

3.1 Vortex wind field

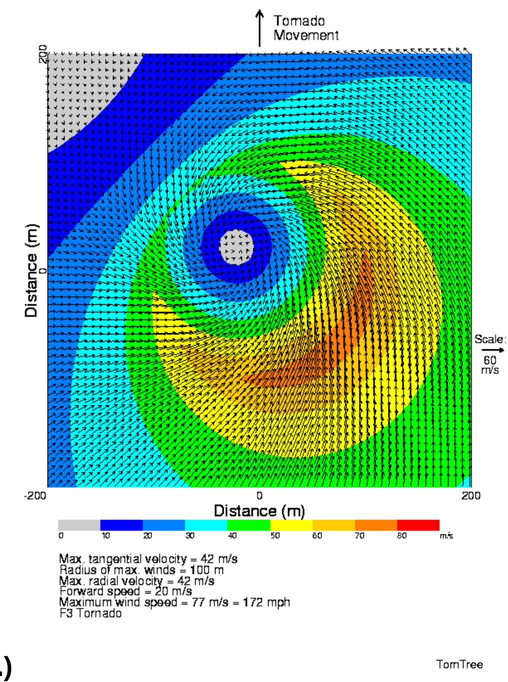
Figure 3.1.1 is a four-panel contour plot showing the differences in the flow pattern when ? and G_{max} are varied. Where G_{max} is the ratio of the maximum rotational wind to the forward speed and ? is the angle measured counterclockwise from a vector pointing toward the vortex center to the relative wind vector. In (a.) and (b.) ? is held constant at -45? where tangential wind speed is equal to radial wind speed and G_{max} is changed. The lower the G_{max} value, the faster the tornado is moving forward for a given rotational speed. Higher forward speed in turn causes the maximum wind speed to be higher, and can be seen in Figure 3.1.1 a. and b. The crescent of highest winds becomes larger and more intense. Also the calm center is moved forward and to the left of the track.

In Figure 3.1.1 c. and d., G_{max} is held constant at 6 to represent a slow moving tornado and ? varies. When ?=-60? tangential wind speed is higher than radial wind speed and vise versa for ?=-30?. When tangential wind speed is greater, the crescent of highest winds is slightly behind and to the right of the center, but when radial wind speed is greater, the crescent is rotated clockwise.

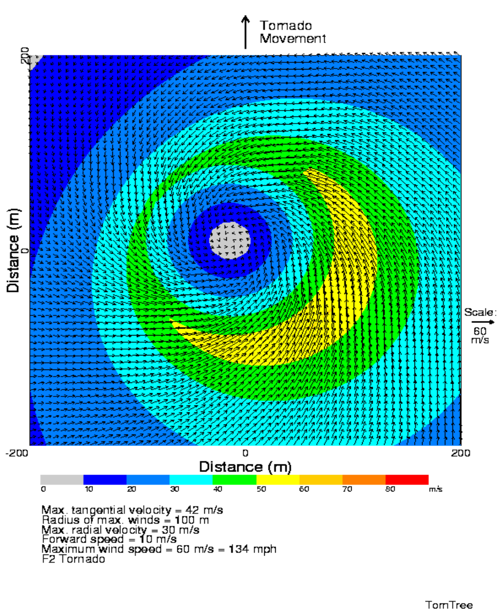
Letzmann (1923) estimated the wind speed at every point in and around a vortex by superimposing a rotational wind field onto a translational one. He then obtained patterns of streamlines and isotachs for the wind field. He found that the flow patterns fell into categories that were determined by G_{max} and ?. Figure 3.1.2a shows streamlines and isotachs for a vortex with a G_{max} of 6 and an ? of -45?. Figure 3.1.2b is a contour plot of the isotachs of a vortex with the same G_{max} and ? produced from the TornTree model. This figure agrees very well with those obtained in hand calculation by Letzmann. The crescent of highest winds is in approximately the same position in the vortex.



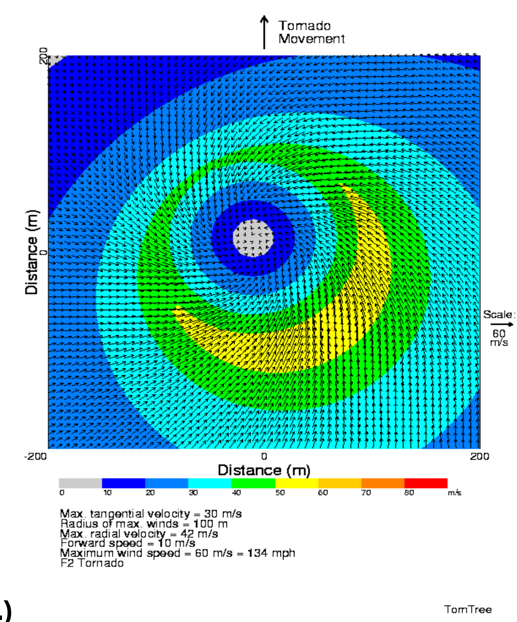
(a.)



(b.)

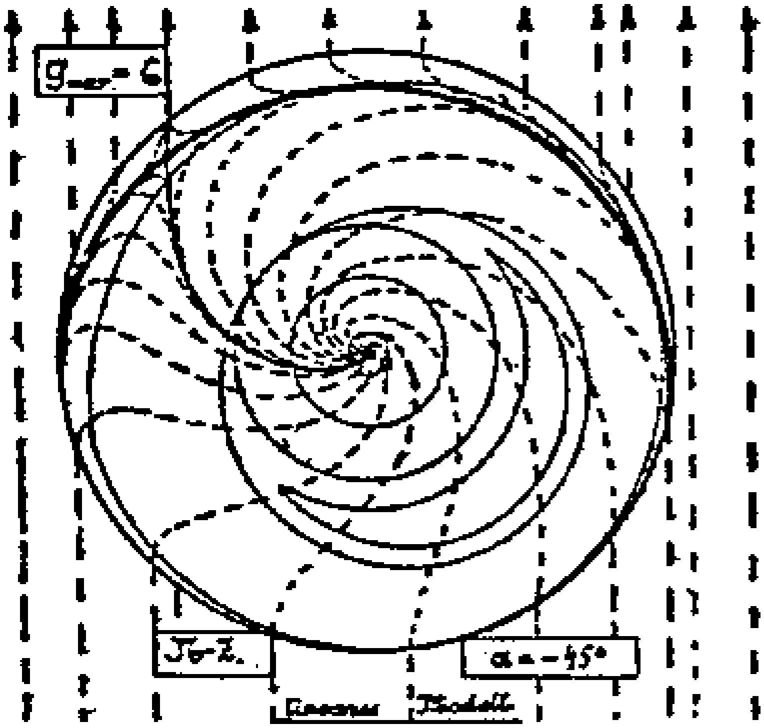


(c.)

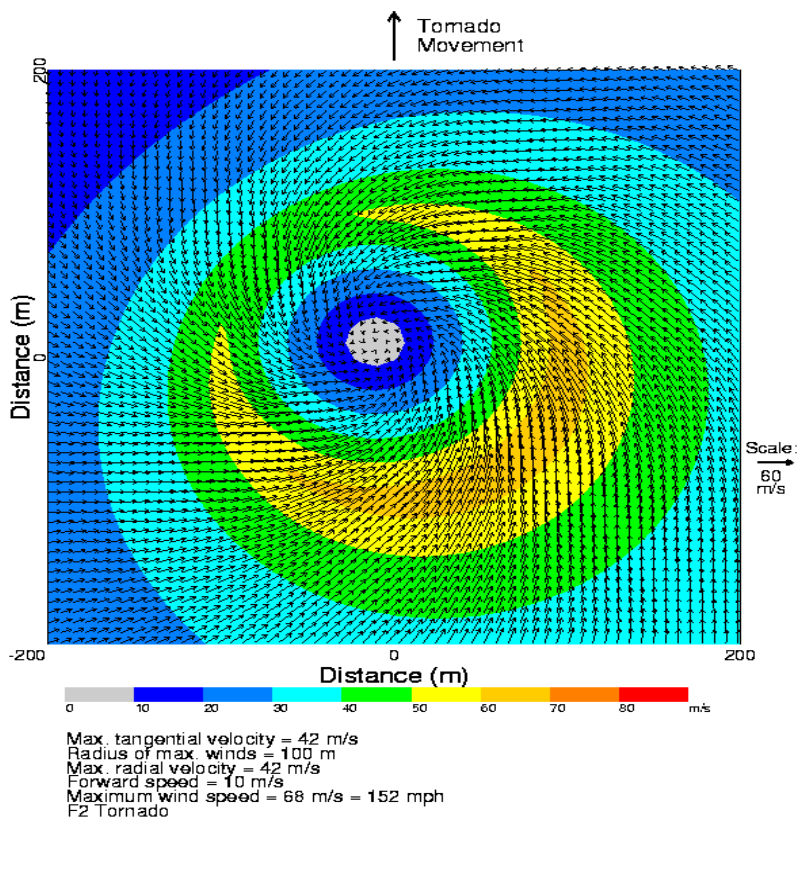


(d.)

Figure 3.1.1: Contour plots from the TornTree model: (a.) G_{max}=6 and ?=-45?, (b.) G_{max}=3 and ?=-45?, (c.) G_{max}=6 and ?=-60?, (d.) G_{max}=6 and ?=-30?.



(a.)



(b.)

Figure 3.1.2: Wind speed contours for vortex with G_{max}=6 and ?=-45?. (a.) from Letzmann (1923), (b.) from the TornTree model.

3.2 Critical wind speeds

3.2.1 Scots pine

In order to error check the TornTree model, characteristics of Scots pine trees were used in the model. After the model is run (only for stem breakage), one can compare the wind speeds that fell trees in the TornTree model to those in the HWIND model.

The data in Table 2 were obtained by running the HWIND model for a Scots pine inside a forest. The HWIND model calculates the uprooting and stem breakage wind speeds for each age class. The TornTree model was also run using Scots pine data to obtain the critical wind speeds. These wind speeds are not adjusted for the gustiness, but represent the above stand mean wind speeds, which are 85% of the gusting wind speeds. These data are also shown in Figure 3.2.1.

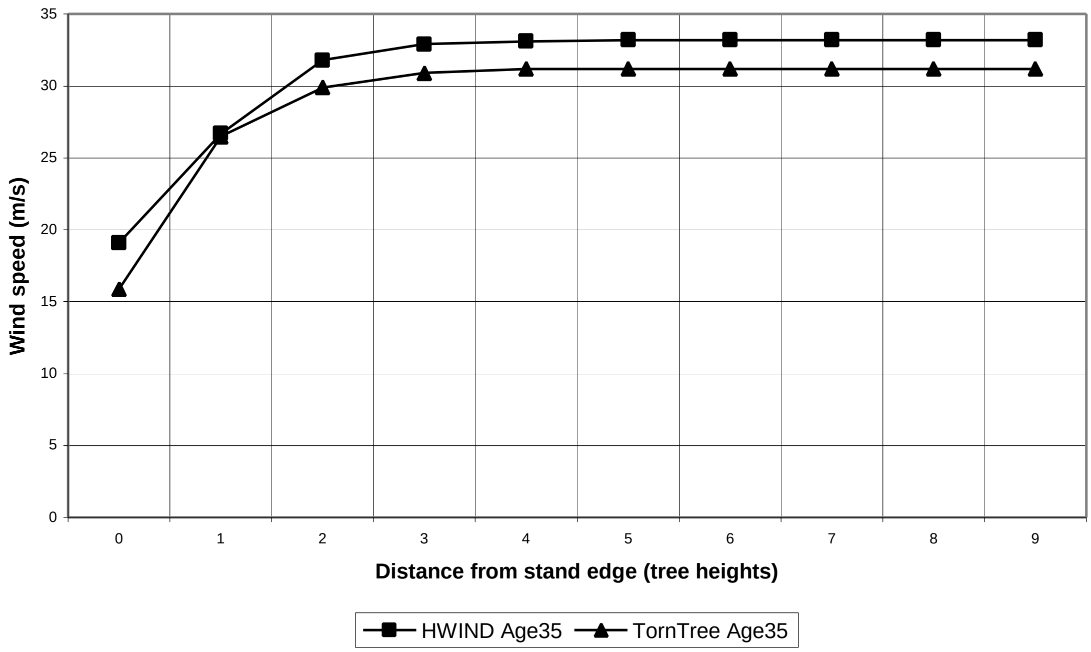
Table 3.2.1. Critical wind speeds (m/s) for stem breakage of Scots pine as a function of distance from stand edge in tree heights, h.

		0h	**1h**	**2h**	**3h**	**4h**	**5h**
Age 35	**HWIND**	19.1	26.7	31.8	32.9	33.1	33.2
	TornTree	15.1	26.3	30.0	31.2	31.5	31.6
Age 65	**HWIND**	21.2	29.4	34.4	35.6	35.8	35.9
	TornTree	14.6	28.3	33.4	34.8	35.1	35.2
Age 95	**HWIND**	22.2	33.7	39.0	40.4	40.6	40.7
	TornTree	20.2	32.2	37.8	39.3	39.7	39.7
Age 120	**HWIND**	23.8	35.9	41.3	42.7	43.0	43.1
	TornTree	23.1	34.9	40.8	42.4	42.7	42.8

Critical Wind Speed:

HWIND vs. TornTree

(a.)



(b.)

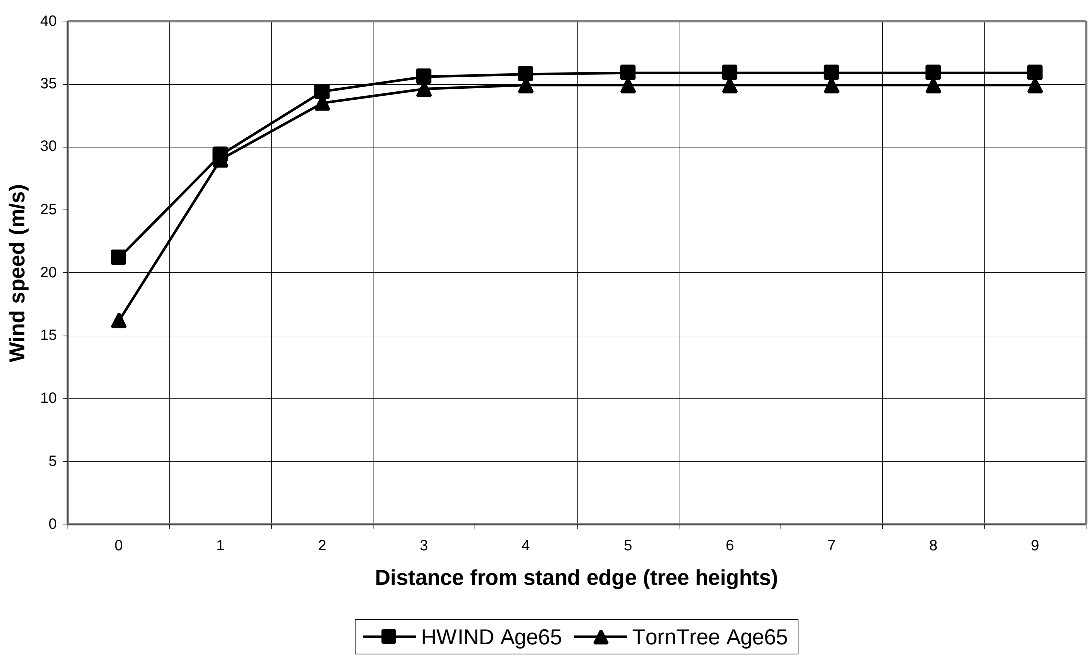
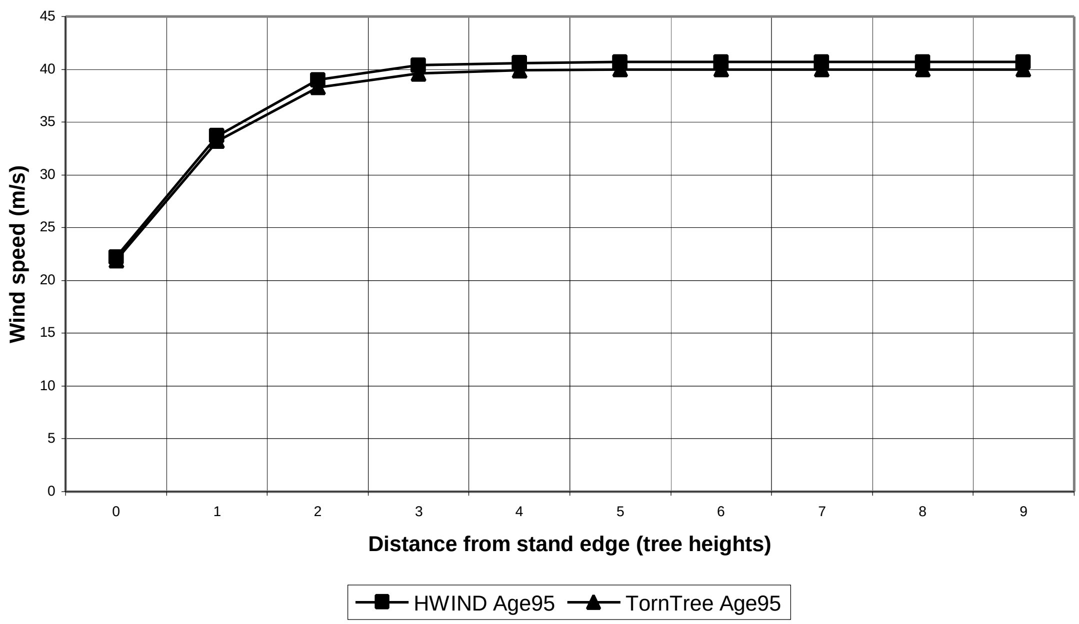


Figure 3.2.1: Critical wind speed for stem breakage of Scots pine with distance into the stand for all age groups (*con't.*).

(c.)



(d.)

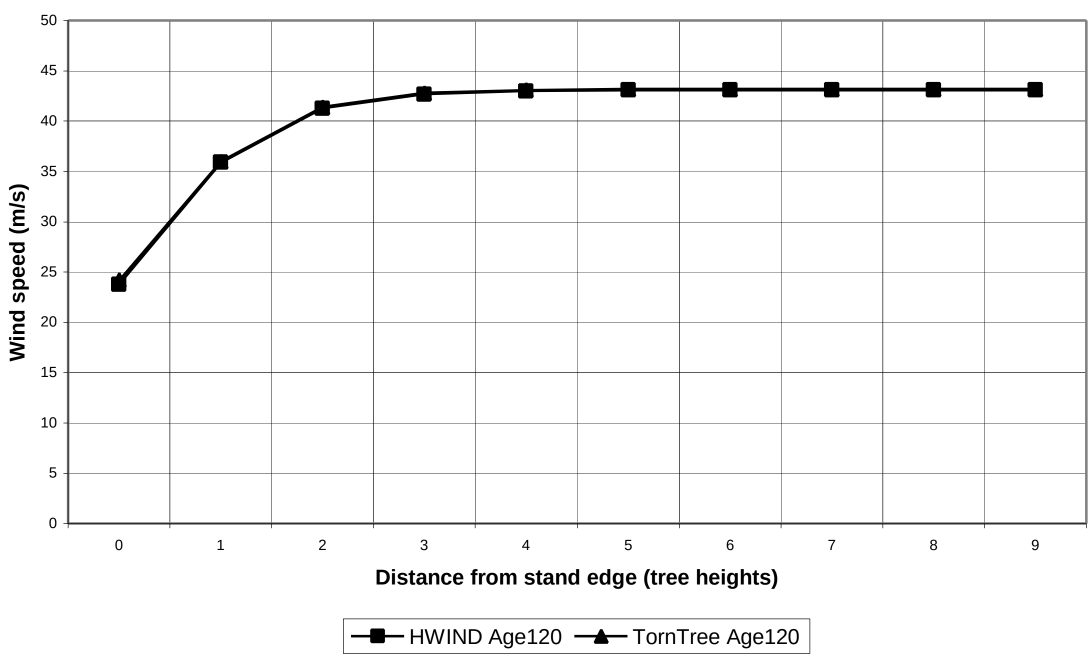


Figure 3.2.1: *con't.*

One factor responsible for the difference in critical wind speeds from the HWIND and TornTree models is crown streamlining. Crown area distribution is zero below the crown and therefore when streamlined is also zero. Data obtained from the HWIND model shows values above zero for streamlined crown area below the crown. Since the TornTree model reads in the unstreamlined crown area and does the streamlining itself, all the values below the crown are zero.

Since the TornTree model is written in Fortran77 and the HWIND model is written in C++, rounding is handled differently and contributes to the small differences in critical wind speeds.

3.2.2 Loblolly pine

The data used for loblolly pine in the TornTree model were taken from Table 25 (Wahlenberg 1960) (see Appendix). This table includes the height, diameter at breast height (DBH) and spacing for trees age 20 to 80 years in Site Indices of 60 to 120. Site Index is determined for a tree species based on the several soil factors relative to that area (Hamilton 1993). The Site Index is the height of 50-year-old dominant trees of the same species in a given Index area. For every Site Index, the TornTree model assumes a random distribution of trees from 20 to 80 years old. The TornTree model calculates the critical wind speed for any inputted height, DBH and spacing.

Figure 3.2.2 shows how the critical wind speed varies with height and taper. Taper is height in meters divided by the DBH in meters. A high taper value indicates a tall, thin tree, while the opposite is true for a low taper. Low taper trees tend to have higher critical wind speeds than high taper trees regardless of height, although, it does appear that shorter trees may have even higher critical wind speeds.

Spacing can also have a large effect on critical wind speed. The model assumes that spacing is equal to crown width. Therefore, tree crowns will meet, but not overlap and will leave no space between trees. The larger the crown the more surface area for the wind to act on, thus increasing the wind force and decreasing critical wind speed. Figure 3.2.3 is a plot of critical wind speed versus spacing for two trees of differing taper. As the spacing increases, crown width increases and critical wind speed decreases.

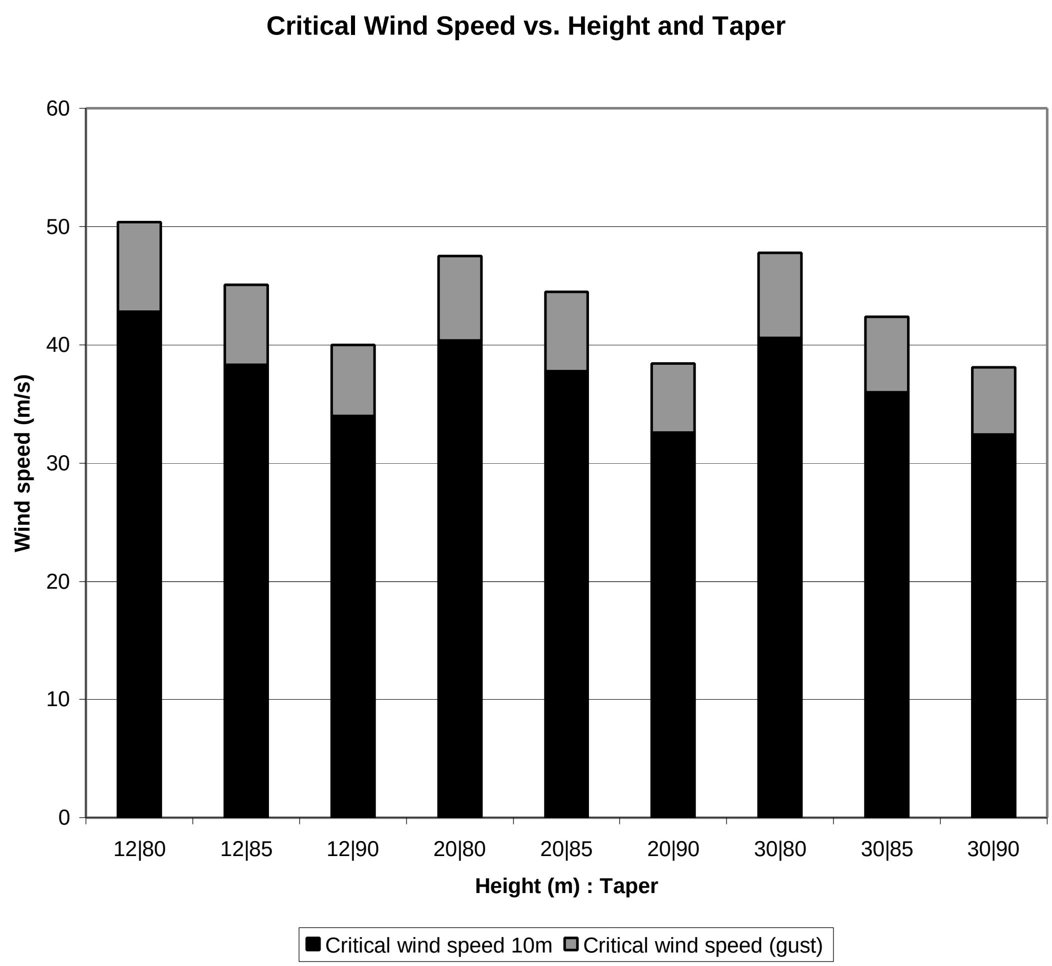


Figure 3.2.2: Critical wind speeds for stem breakage of loblolly pines of differing height and taper (height|taper), where taper is the height in meters divided by the diameter at breast height in meters.

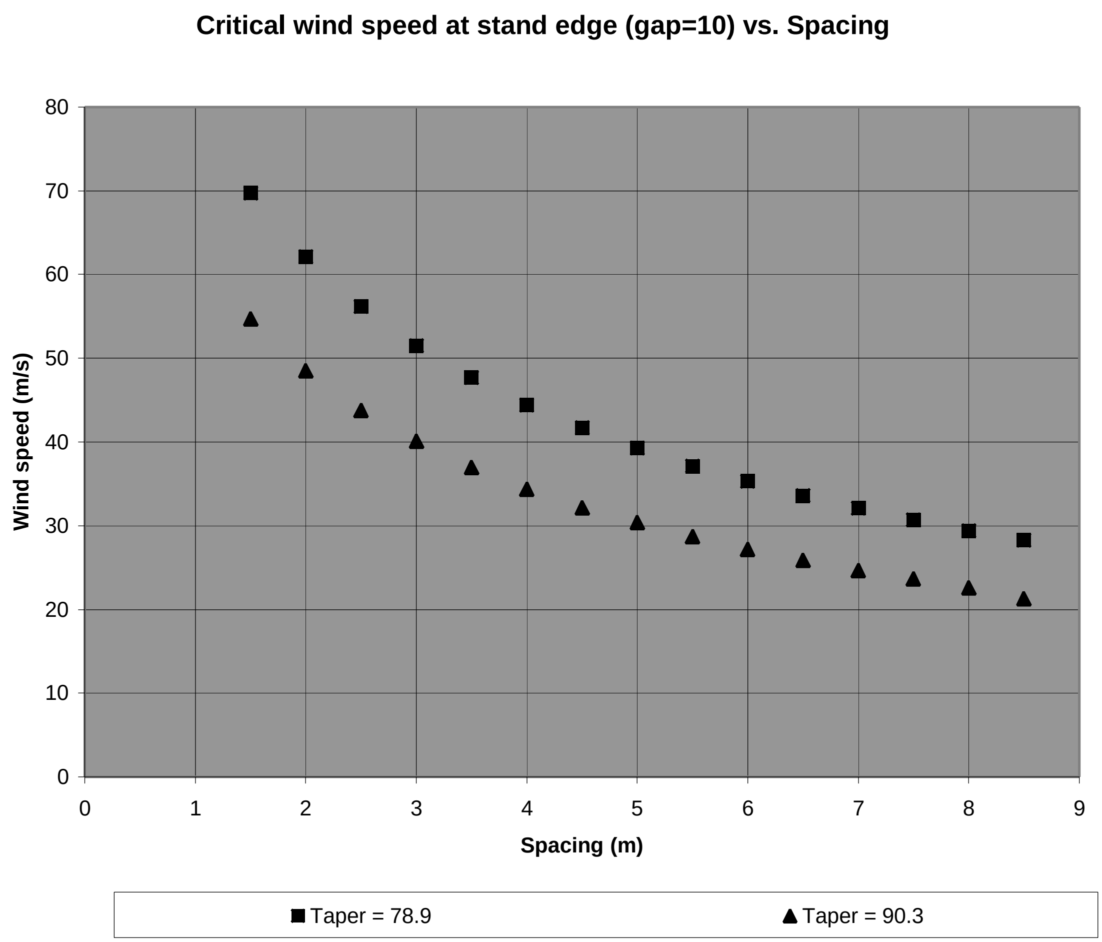


Figure 3.2.3: Critical wind speeds for stem breakage of loblolly pine for different tree spacings for two taper values.

3.3 Tree damage patterns

3.3.1 Tornadoes

The TornTree model outputs a graphical plot of downed trees. The arrows are proportional to the tree height and represent downed trees in the instantaneous direction of the wind, since there is no interaction among the trees. The gray dots represent standing trees. Figures 3.3.1 through 3.3.3 are examples of the model output for loblolly pine of Site Index 90 versus F-number. The tornadoes used in these figures have G_{max} of 4 and ? of approximately -60?. In the case of the F1 tornado (Fig. 3.3.1a), only a few trees at the stand edge are knocked down and are orientated in the direction of tornado motion (Fig. 3.3.1b). The tornado's wind speeds are not strong enough to knock down trees in the stand and only the winds to the right of the center are strong enough to knock down trees at the stand edge. This is because of the addition of the forward motion to the rotational motion of the tornado. When an F3 tornado (Fig. 3.3.2a) is moved through the same forest, the forward motion of the tornado strengthens the winds on the right side of the tornado by adding to the tangential wind speeds, causing the tree-falls to converge to the right of the tornado path (Fig. 3.3.2b). In an F5 (Figs. 3.3.3a), the rotational wind is much higher than the forward motion and trees fall in a curved path with trees falling opposite the direction of tornado motion on the left side (Figs. 3.3.3b).

Letzmann (1923) proposed that different combinations of tangential, radial and forward velocities cannot yield identical damage patterns, even if the maximum wind speeds are the same. The following four tornadoes have equal maximum wind speeds obtained through different combinations of tangential, radial and forward velocity. The radius of maximum winds is held constant throughout Figures 3.3.4 through 3.3.7 since it will only affect the width of the damage path. Radial wind speed is greater than the tangential wind speed in Figure 3.3.4a. The combination of radial wind speed and forward motion at the rear of the tornado causes the trees to fall in the direction of tornado motion (Figure 3.3.4b). This pattern is similar to the pattern of a Type III, Form b pattern from Letzmann's calculations (see Fig. 1.3.3). Tangential and radial wind speeds are equal in Figure 3.3.5a. This damage pattern (Figure 3.3.5b) looks very similar to Figure 3.3.4b, but upon close inspection one finds that the angle of tree fall is slightly more in the direction of motion in Figure 3.3.5b. When tangential wind speed is higher than the radial wind speed, as in Figure 3.3.6a, the trees fall even more towards the center of the tornado track (Figure 3.3.6b). A well-defined line of convergence of fallen trees forms just to the right of the center of the tornado path. This pattern can be compared to Letzmann's calculations of a Type III, Form c vortex (Fig. 1.3.3). Figure 3.3.7a is the same as the previous tornado with a higher forward speed. This causes the trees to fall more in the direction of tornado motion than towards the track center (Figure 3.3.7b).

To show how the tree fall patterns differ with Site Index, an F2 tornado (Figure 3.3.8) was used. The patterns show many similarities (Figures 3.3.9 – 3.3.11). They all show an area of convergence to the right side of the tornado track. The only striking difference is that as Site Index increases, the number of downed trees decreases. This is caused by the fact that with a lower Site Index the trees are closer together since in the model, spacing increases with Site Index, therefore, there are more trees in the path of the strongest winds.

3.3.2 Microbursts

The TornTree model is also capable of generating a microburst and moving it through a forest. If tangential wind speed is set to zero and the radial wind speed is negative, i.e., outflow, then the wind field will radiate outward from the center. Figure 3.3.12b is the tree fall pattern associated with the microburst in Figure 3.3.12a. It is a weak microburst, so few trees are knocked down, but the pattern does show the effect of the divergent wind field. If the maximum microburst velocity is increased, the overall wind speed increases (Figure 3.3.13a) and more trees are knocked down (Figure 3.3.13b). If the thunderstorm that produced the microburst in Figure 3.3.12a were to have a higher forward speed (Figure 3.3.14a and b), the pattern is very similar to that of Figure 3.3.13b. All of these microburst patterns are similar to those of Letzmann's calculations (Form f in Fig. 1.3.3).

3.3.3 Tree fall comparison

The objective of this project was to develop a model that will produce plots of damage patterns when given a number of inputs. These plots can then be used to estimate the near surface wind speed and the F-number. To do this, one would need an aeiral photograph of a damaged forest. The forward speed of a tornado is usually known and the radius of maximum winds can be easily estimated from the photograph. One would also need to know the tree spieces and the approximate age or Site Index of the stand. By comparing the photograph to Figure 1.3.3 one can get an estimate of G_{max} and ?. These estimates can lead to a good first guess of tangential and radial wind speeds for input into the model.

Once the model has created the damage pattern plot, one can compare it to the photograph of actual damage. The model user has to decide whether or not the model plot is a good representation of what actually happened. This optical comparison technique allows for adequate estimation of maximum wind speed and F-number, but is inadequate when trying to estimate the tangential and radial velocities that lead to the maximum wind speed (i.e. Figures 3.3.4b and 3.3.5b).

If the user knows the percent of fallen trees in the stand, this can be compared with the value on the model plots to estimate maximum wind speed and F-number.

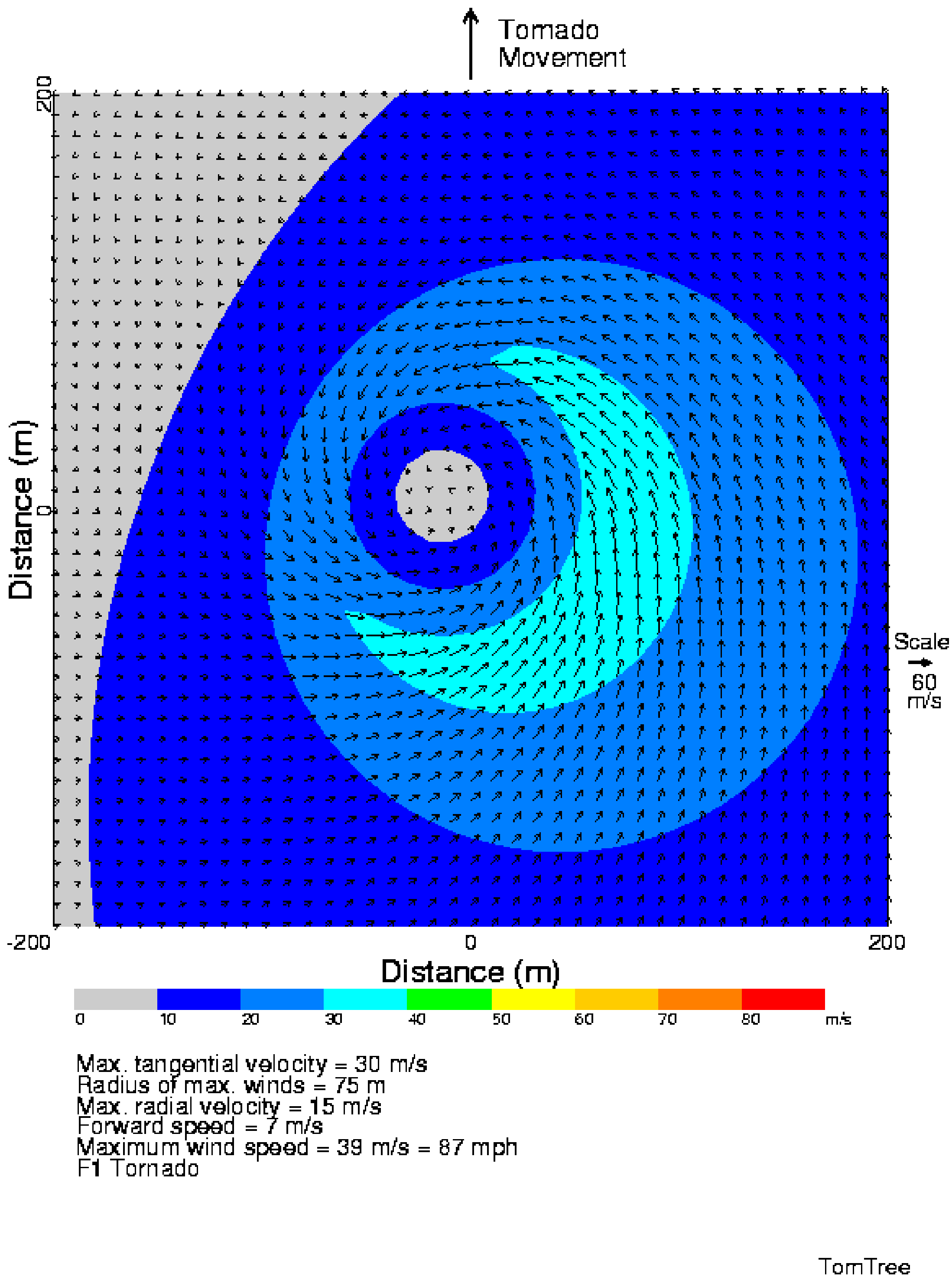


Figure 3.3.1a: Contour plot of an F1 tornado.

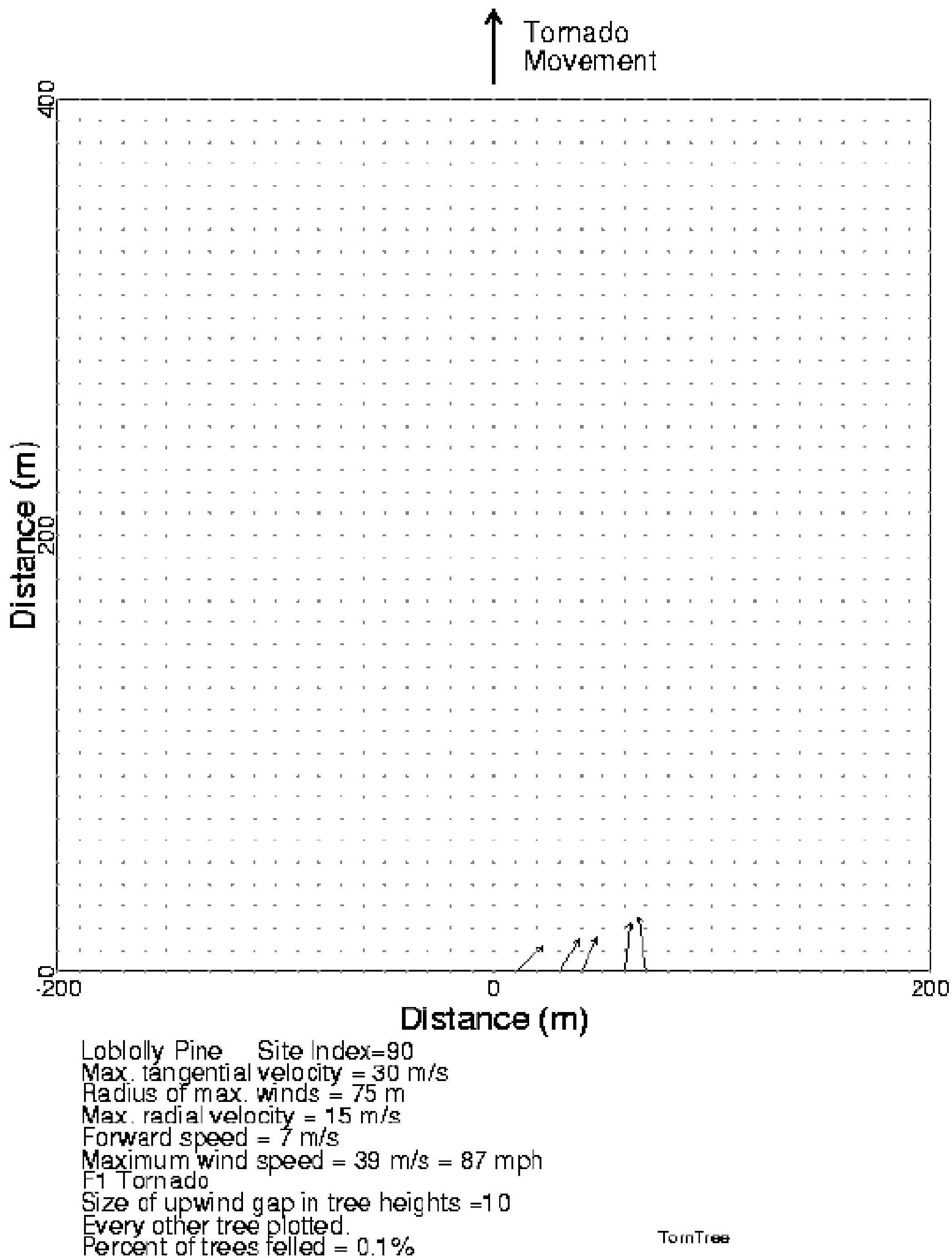


Figure 3.3.1b: Tree fall pattern for the tornado in Figure 3.3.1a.

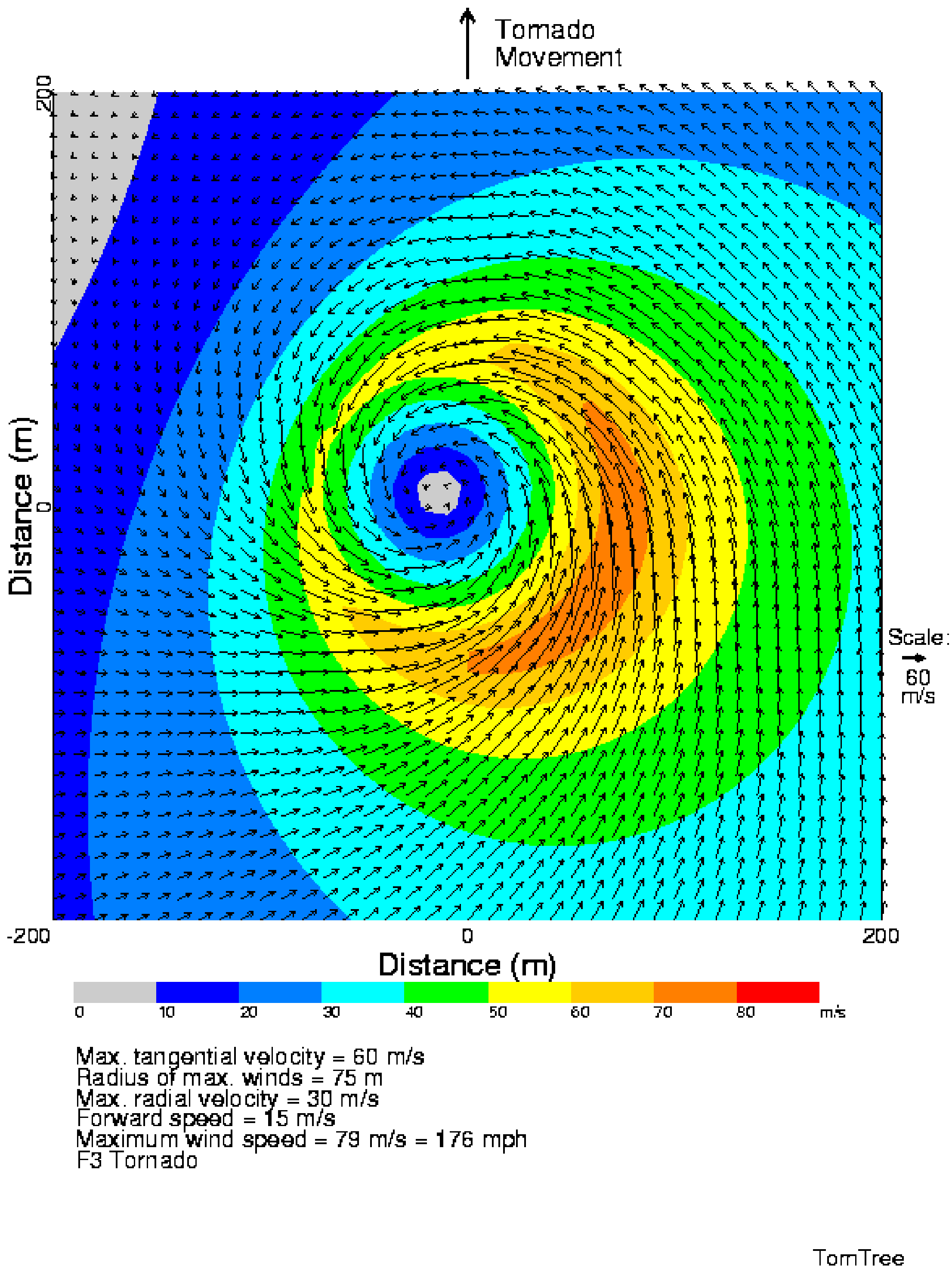


Figure 3.3.2a: Contour plot of an F3 tornado.

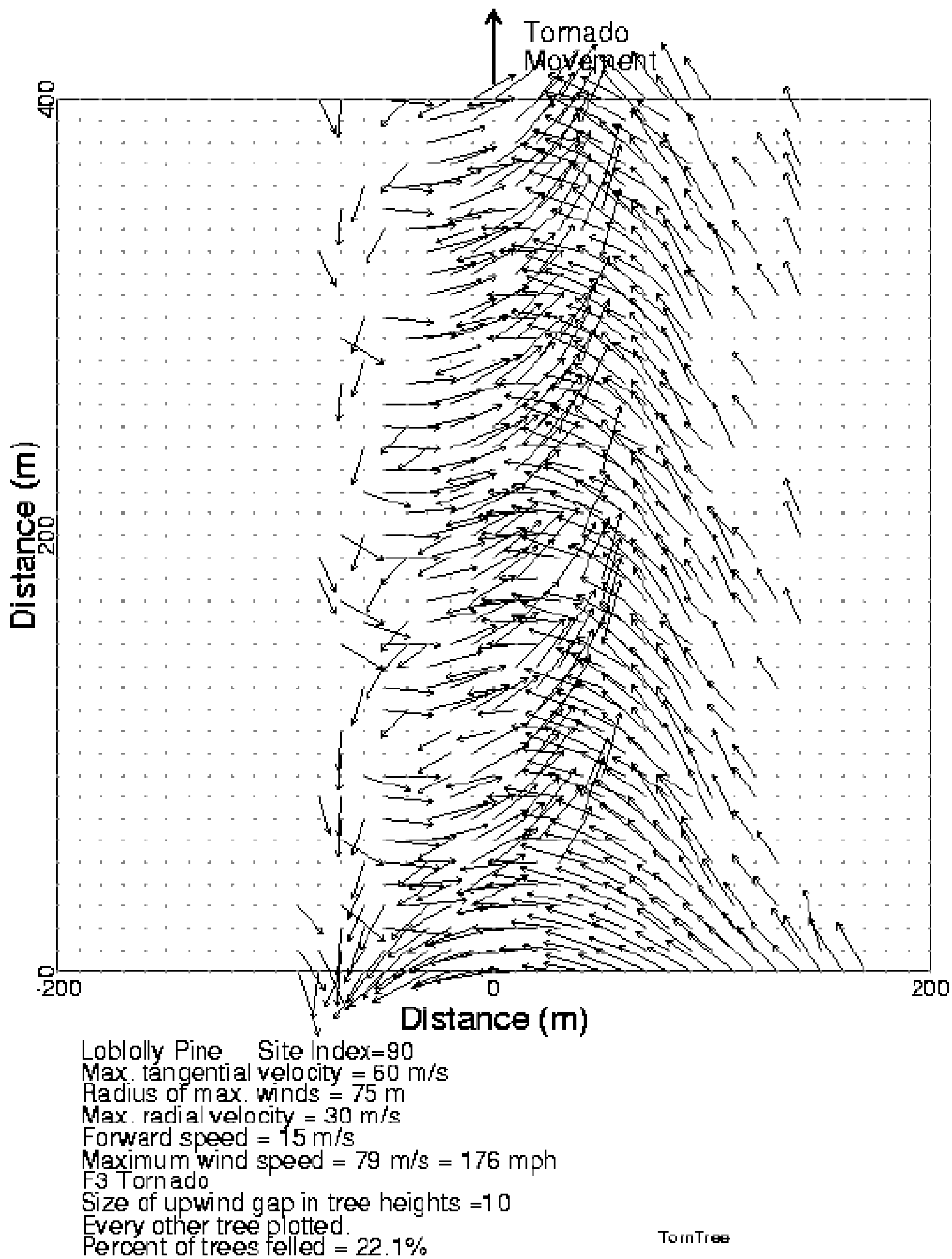


Figure 3.3.2b: Tree fall pattern for the tornado in Figure 3.3.2a.

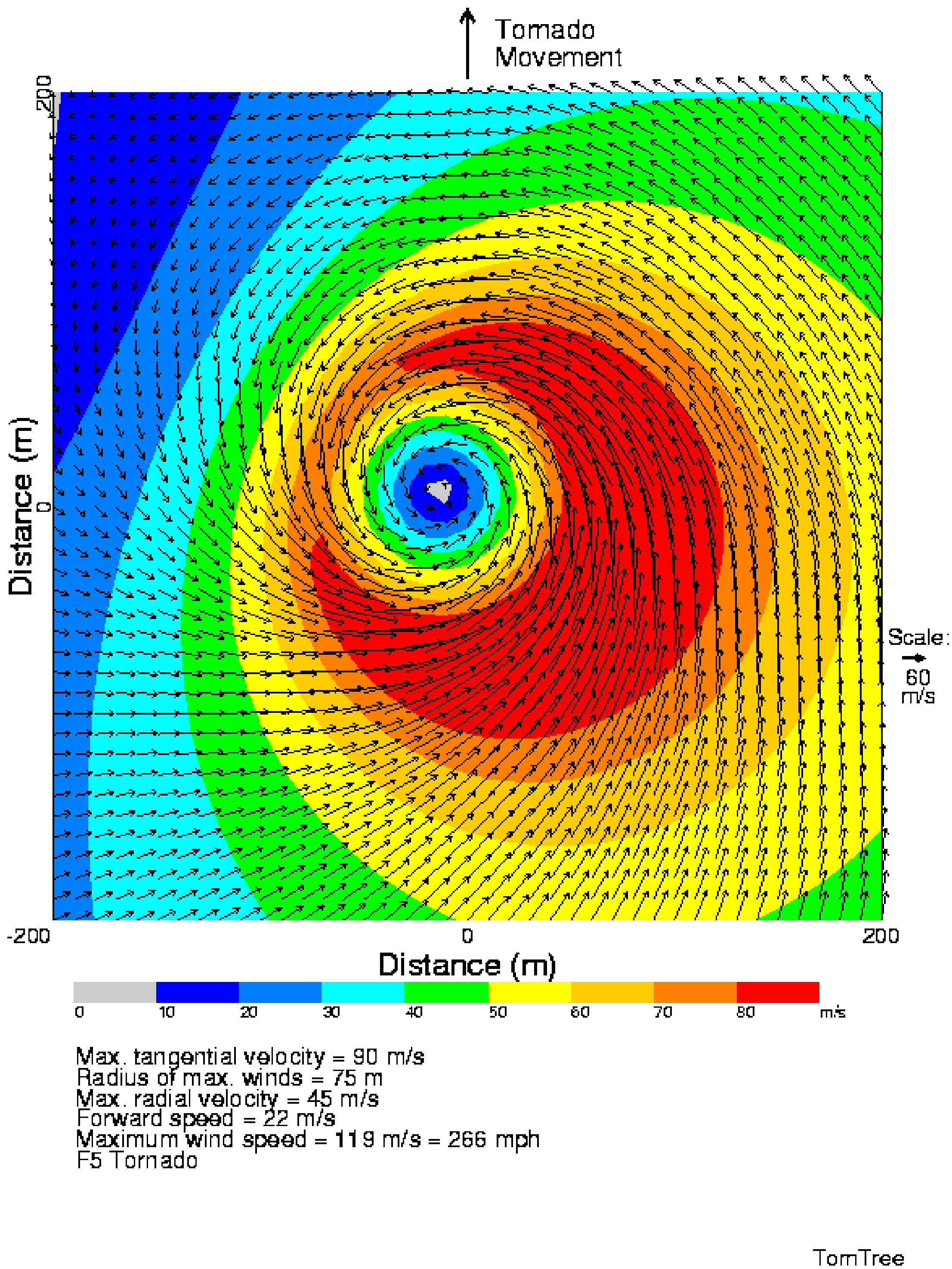


Figure 3.3.3a: Contour plot of an F5 tornado.

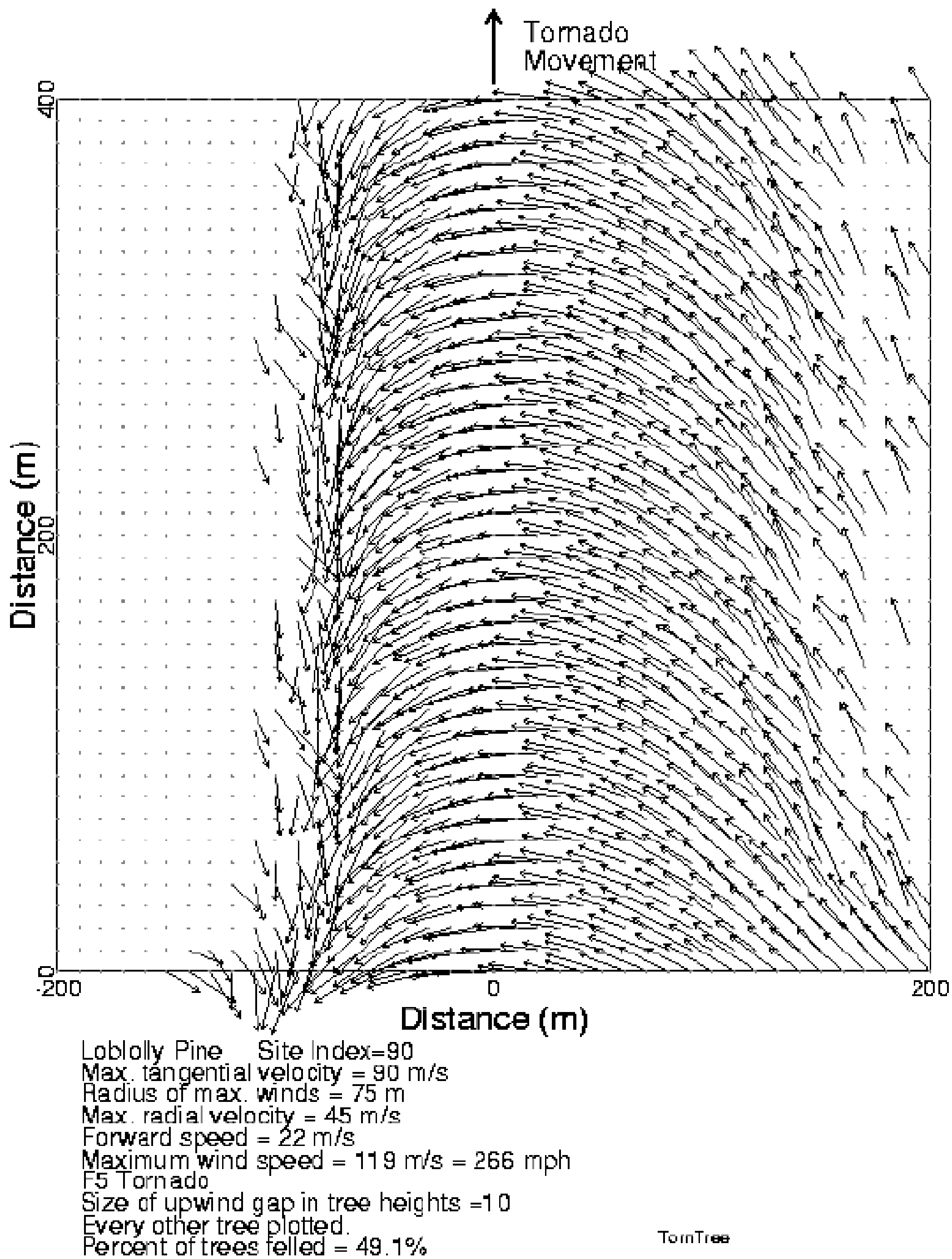


Figure 3.3.3b: Tree fall pattern for the tornado in Figure 3.3.3a.

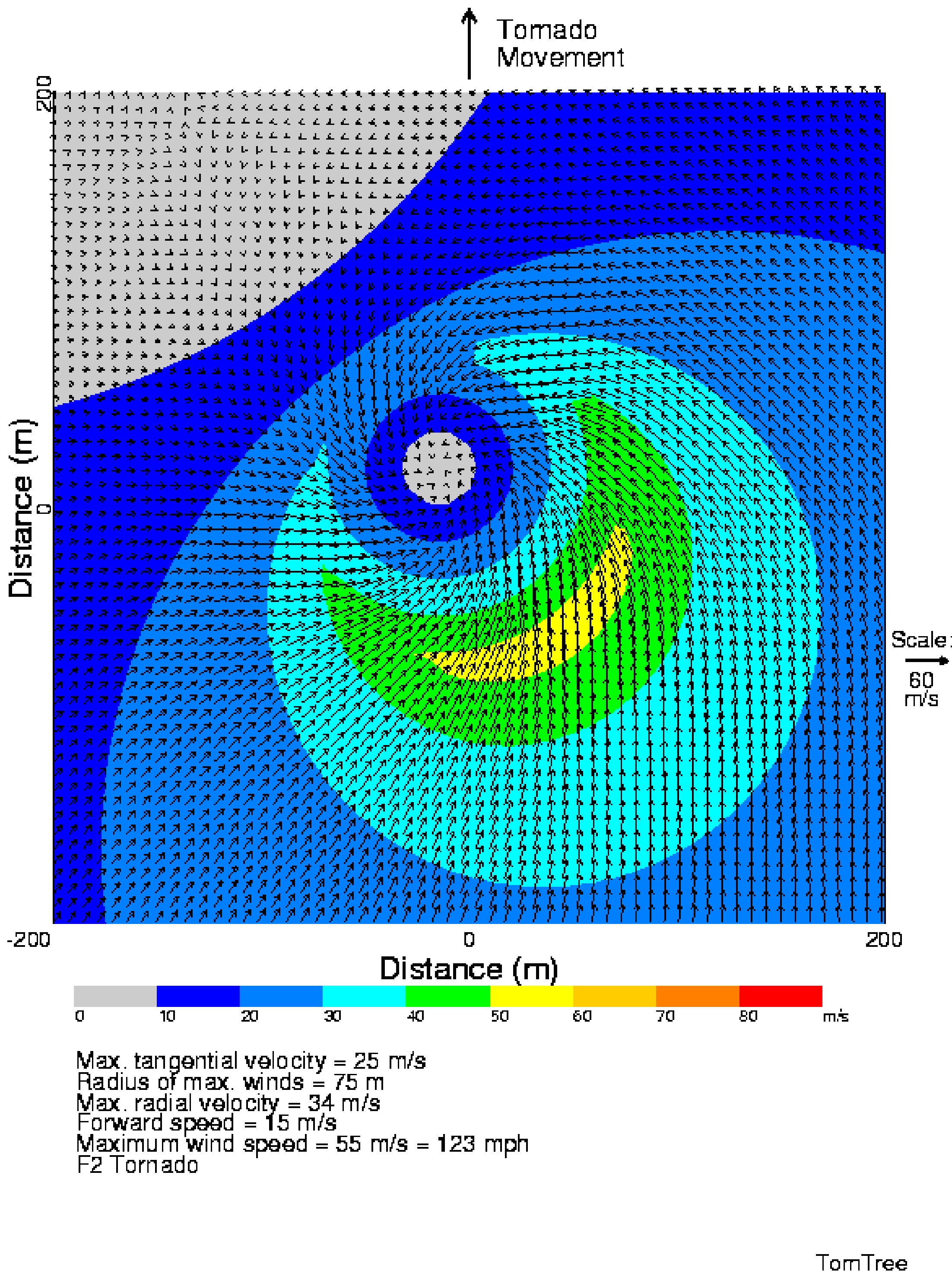


Figure 3.3.4a: Contour plot of tornado with radial wind speed greater than tangential wind speed and slow moving.

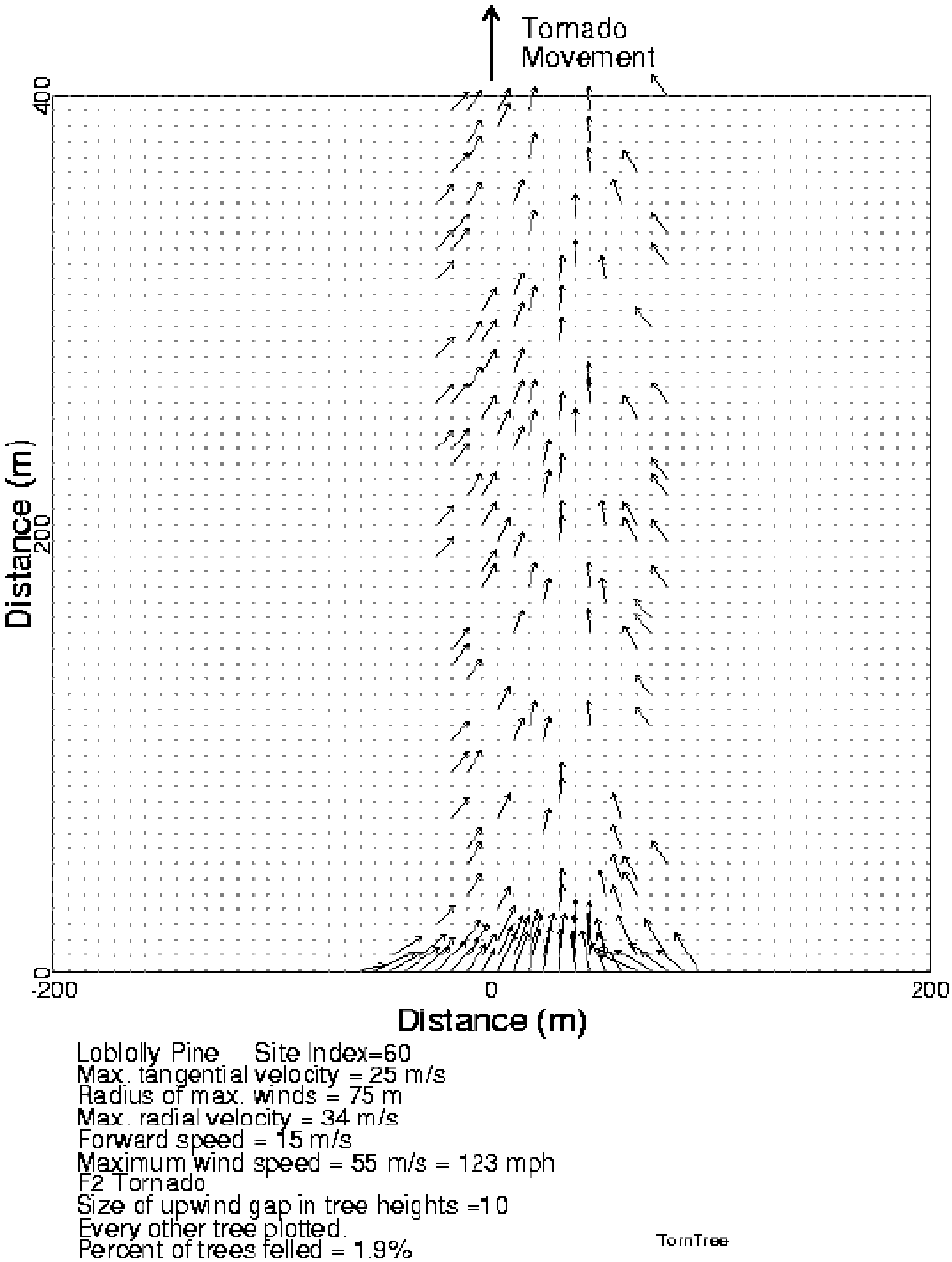


Figure 3.3.4b: Tree fall pattern for tornado in Fig. 3.3.4a.

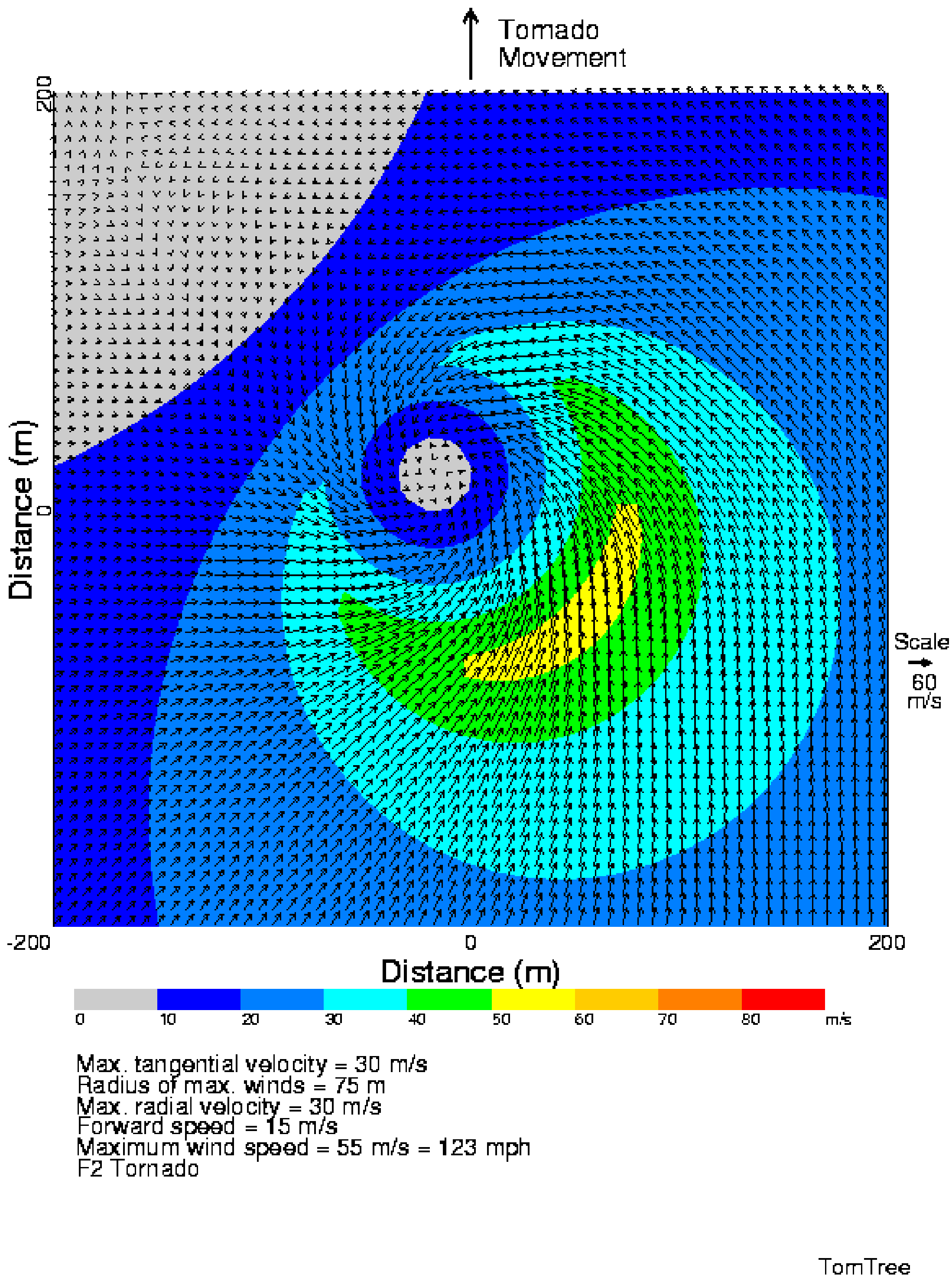


Figure 3.3.5a: Same as Fig. 3.3.4a, with tangential and radial wind speeds equal.

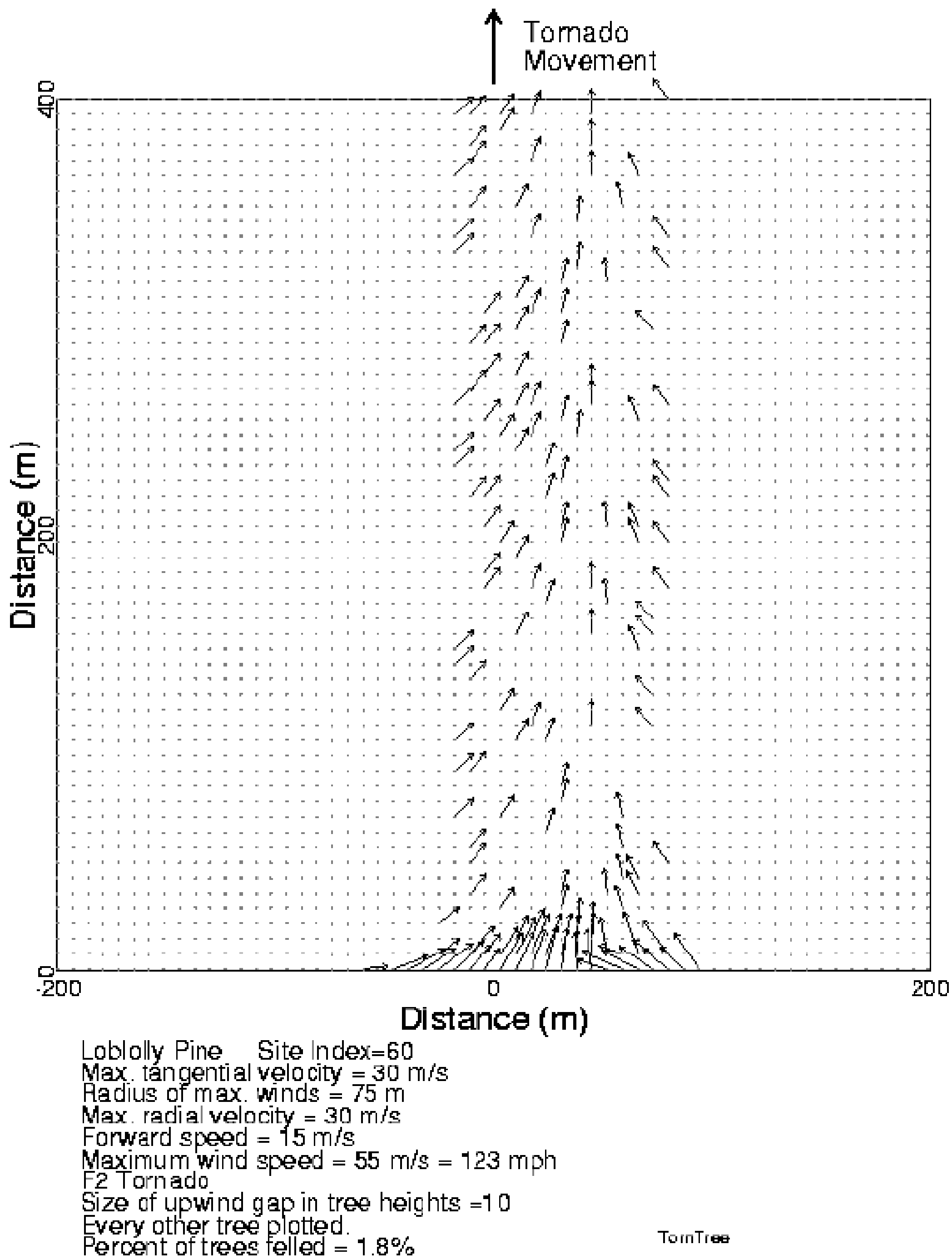


Figure 3.3.5b: Tree fall pattern for tornado in Fig. 3.3.5a.

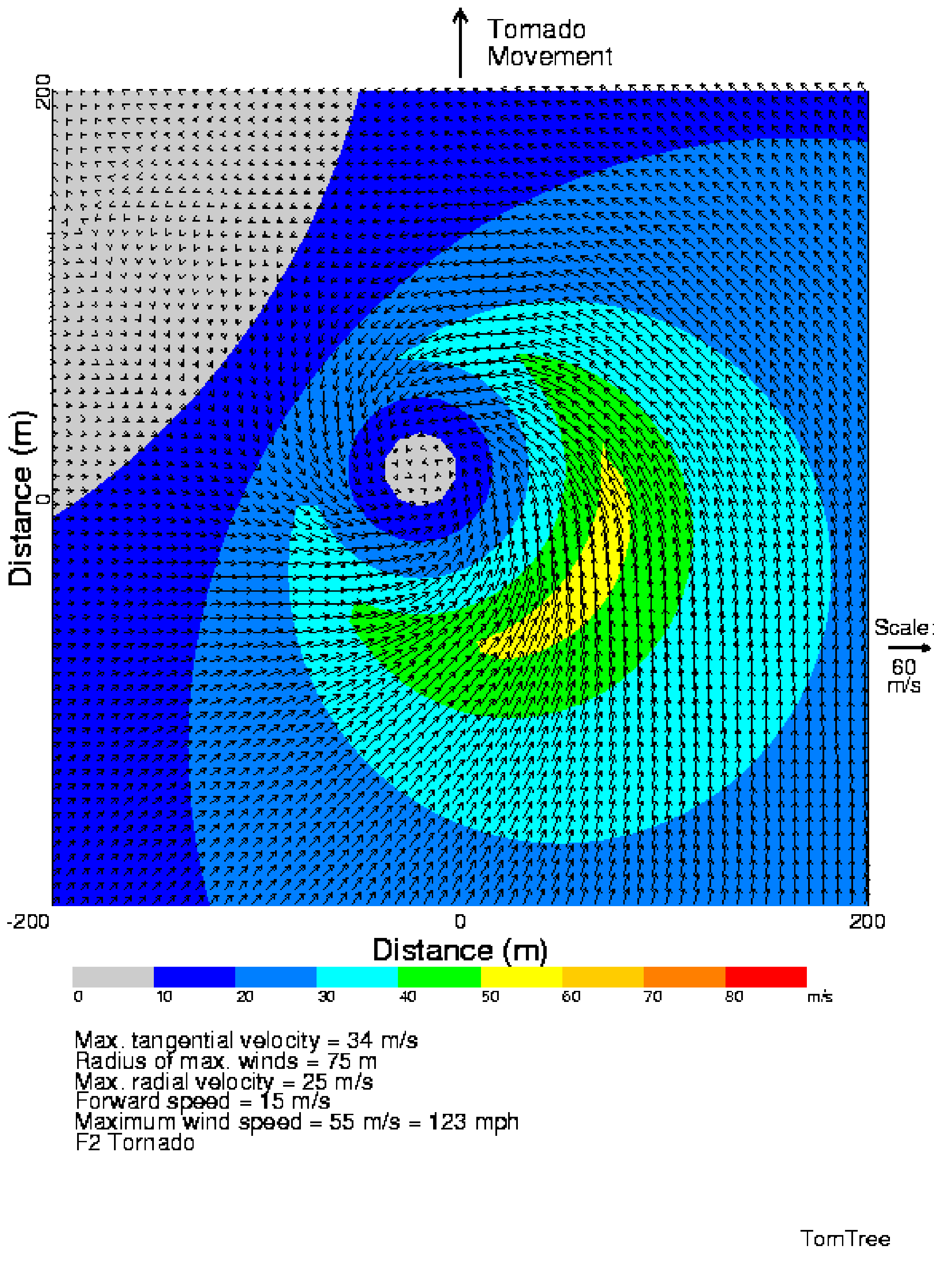


Figure 3.3.6a: Same as Fig. 3.3.4a, with tangential wind speed greater than radial wind speed.

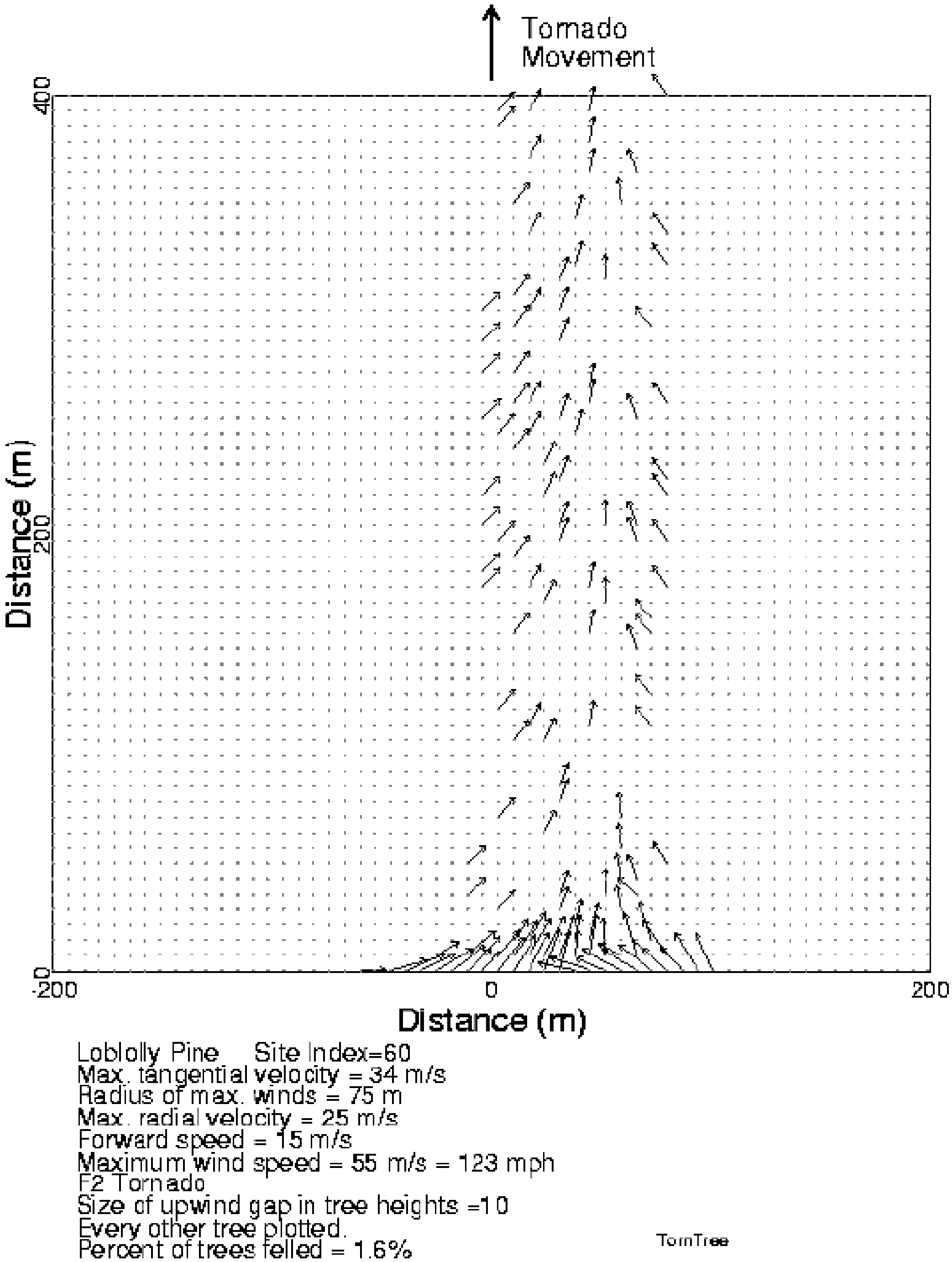


Figure 3.3.6b: Tree fall pattern for tornado in Fig. 3.3.6a.

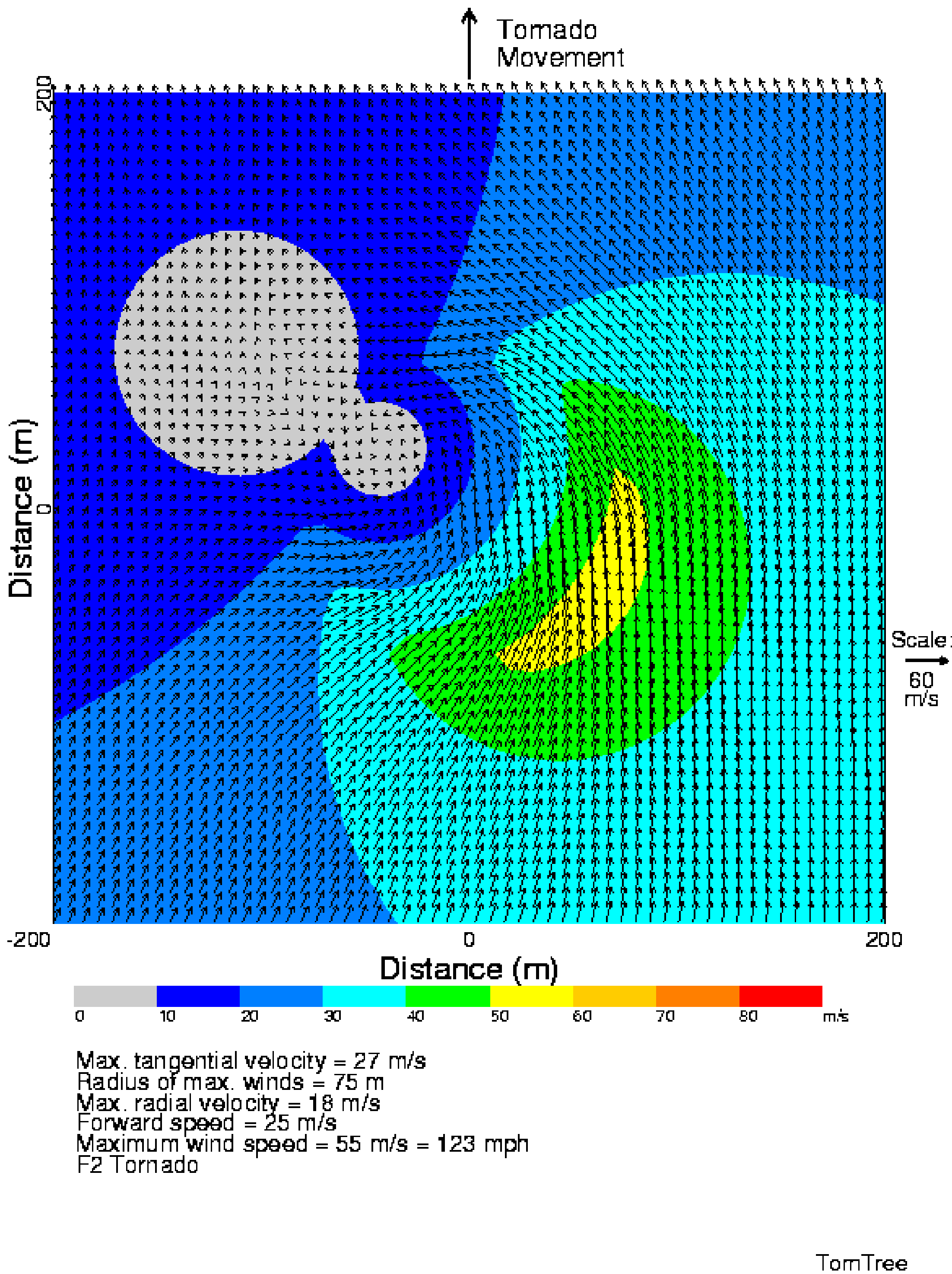


Figure 3.3.7a: Same as Fig. 3.3.6a, with higher forward speed.

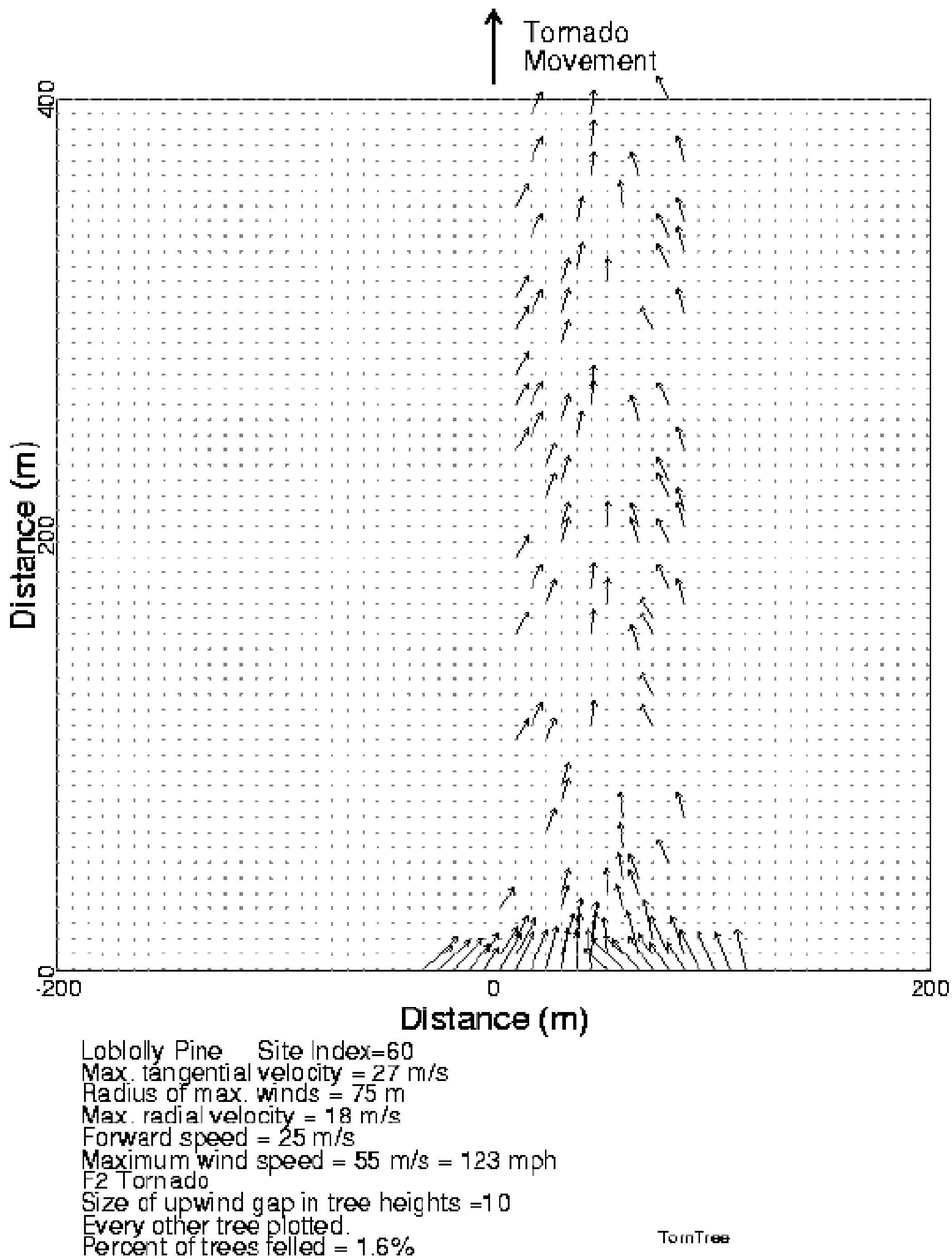


Figure 3.3.7b: Tree fall pattern for tornado in Fig. 3.3.7a.

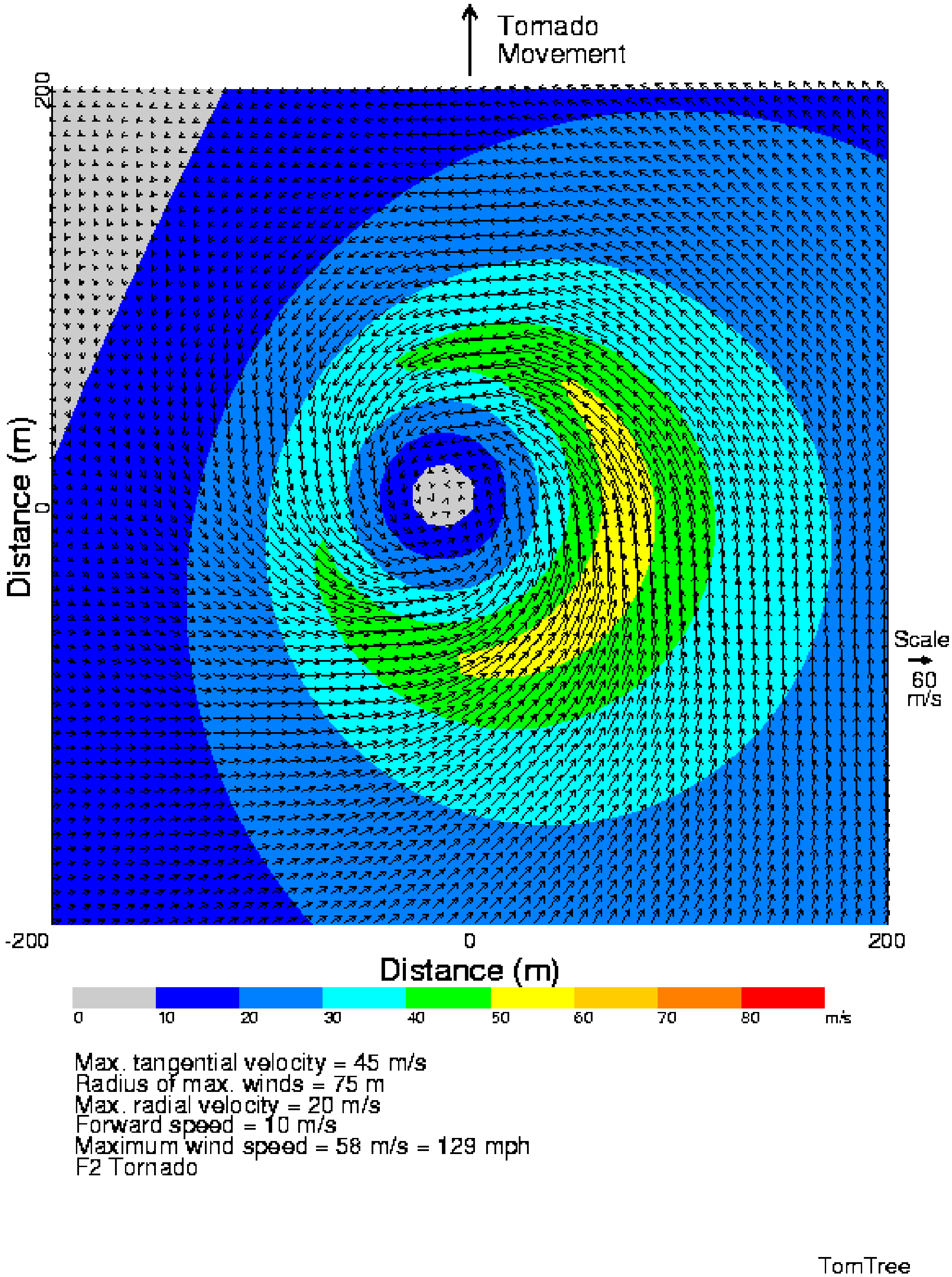


Figure 3.3.8: Contour plot for an F2 tornado used to show differences in tree fall patterns between Site Index.

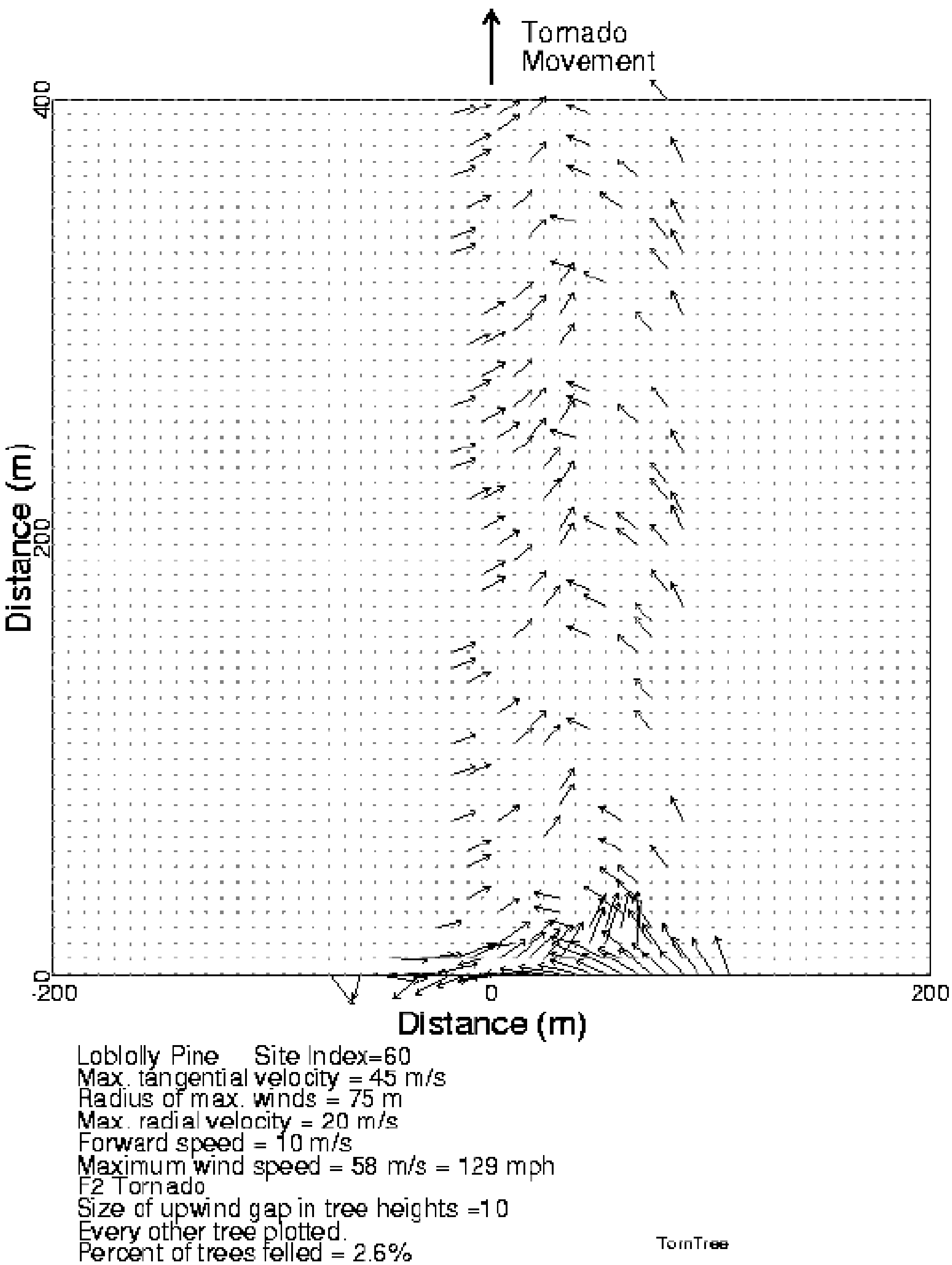


Figure 3.3.9: Tree fall pattern for Site Index of 60 from the tornado in Fig. 3.3.8.

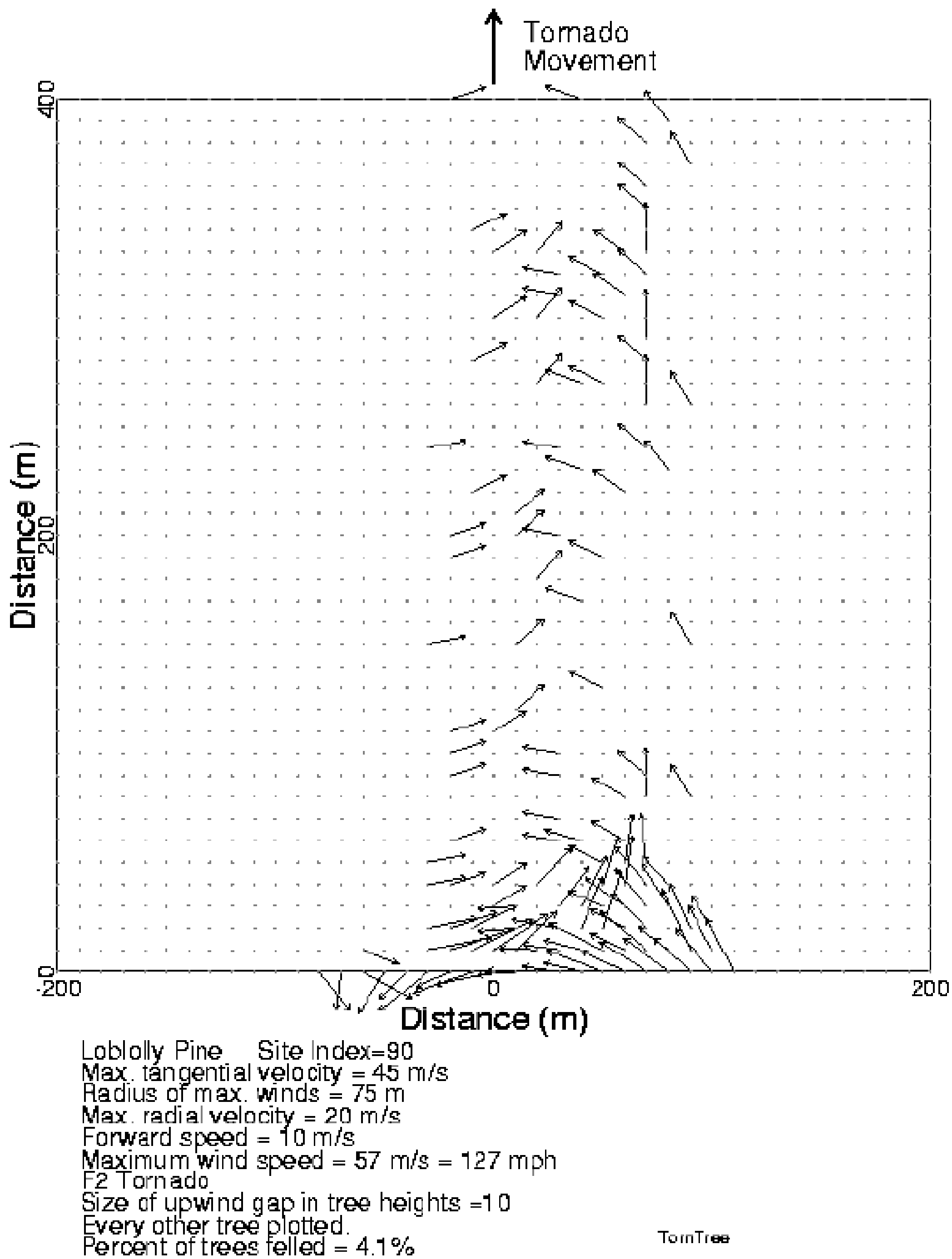


Figure 3.3.10: Same as Fig. 3.3.9, with Site Index of 90.

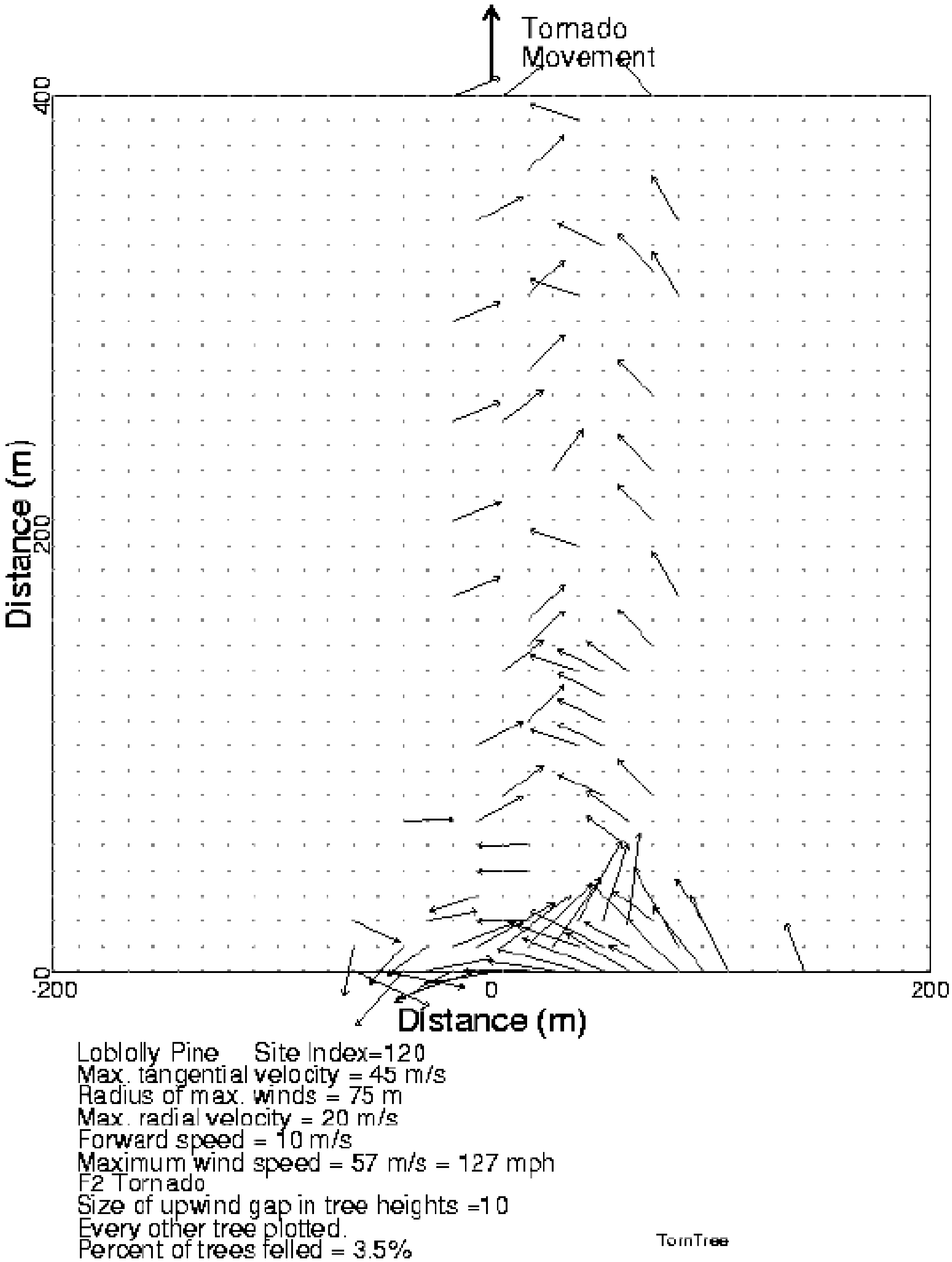


Figure 3.3.11: Same as Fig. 3.3.9, with Site Index of 120.

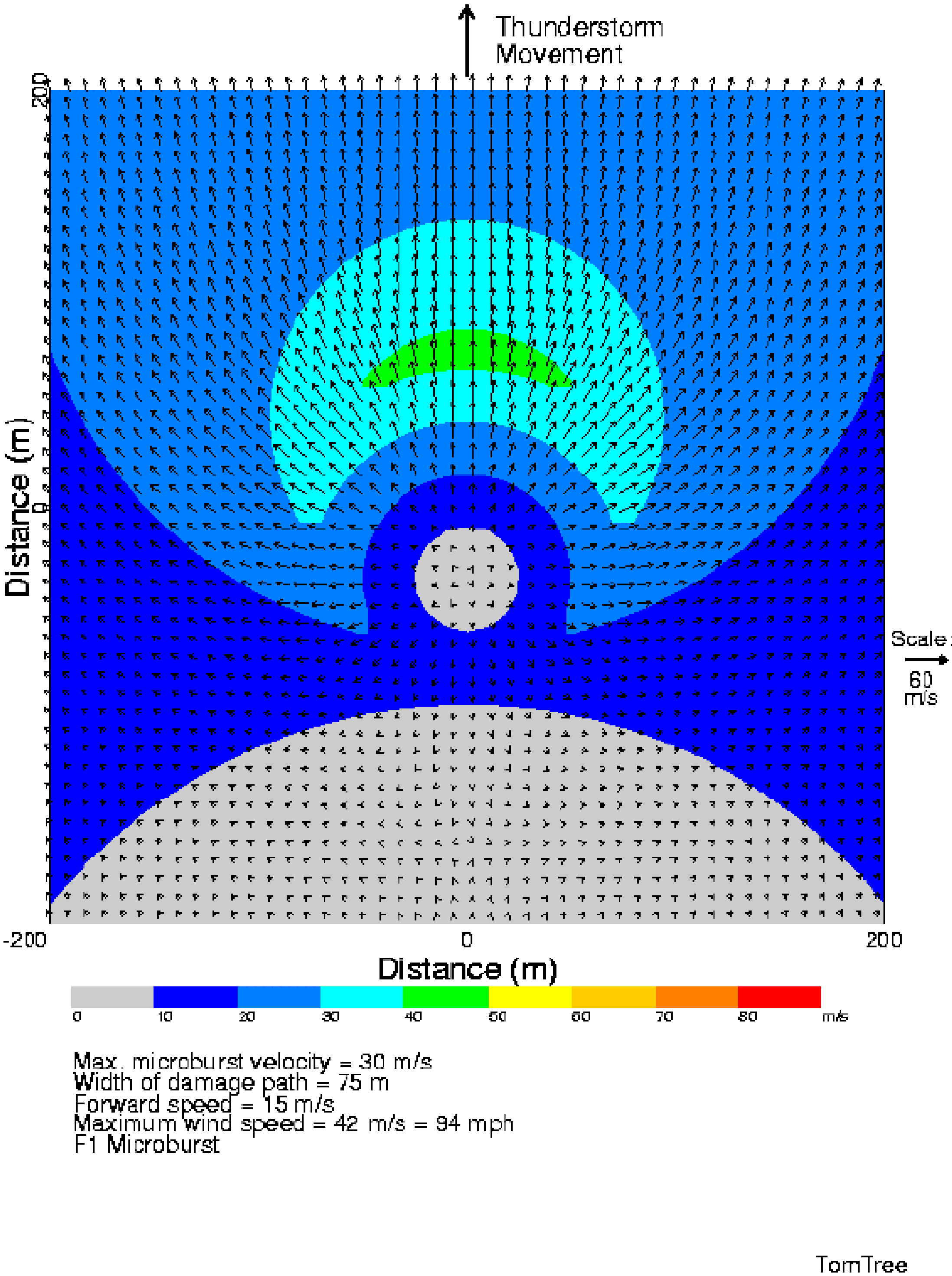


Figure 3.3.12a: Contour plot of a microburst with a low maximum microburst velocity.

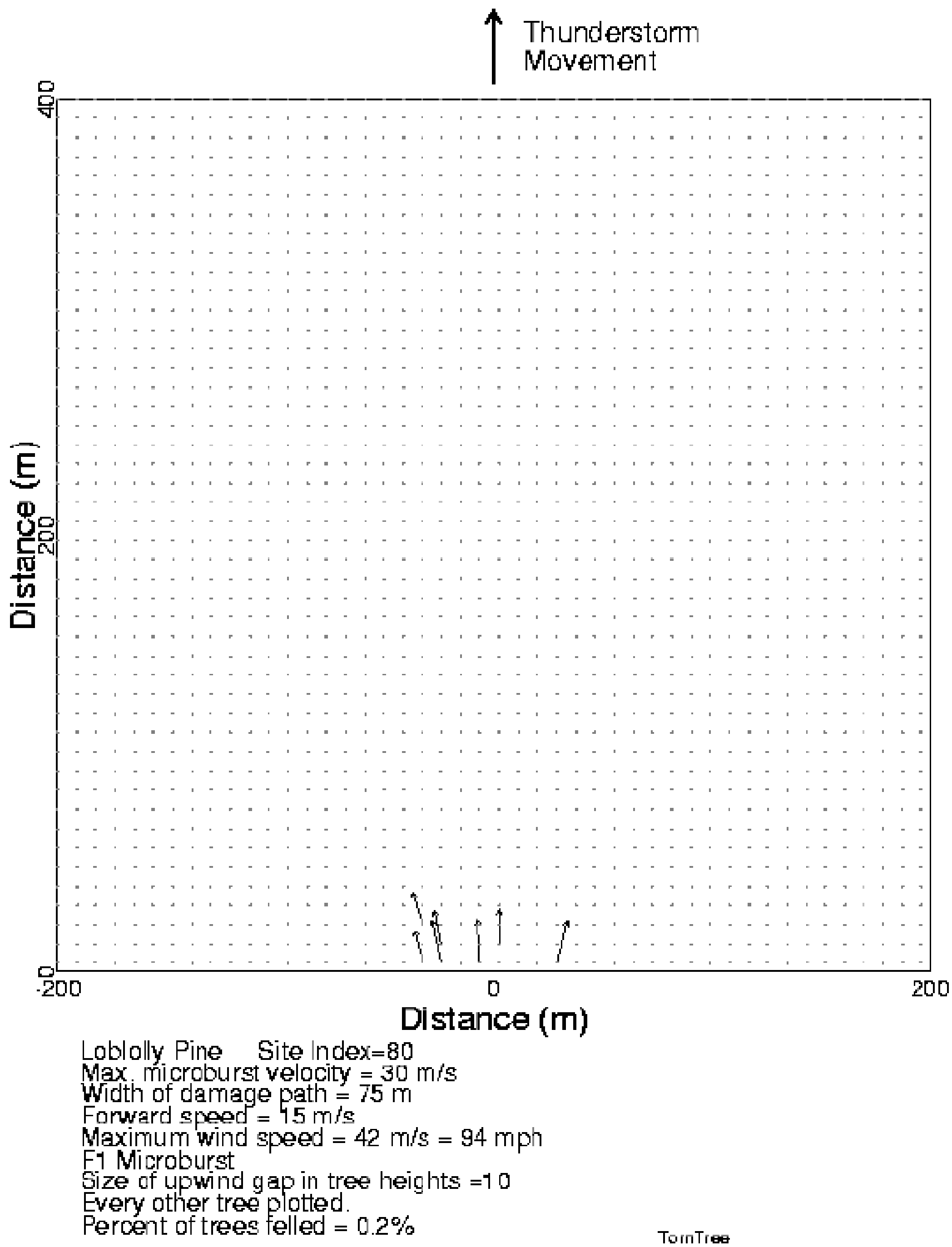


Figure 3.3.12b: Tree fall pattern for microburst in Fig. 3.3.12a.

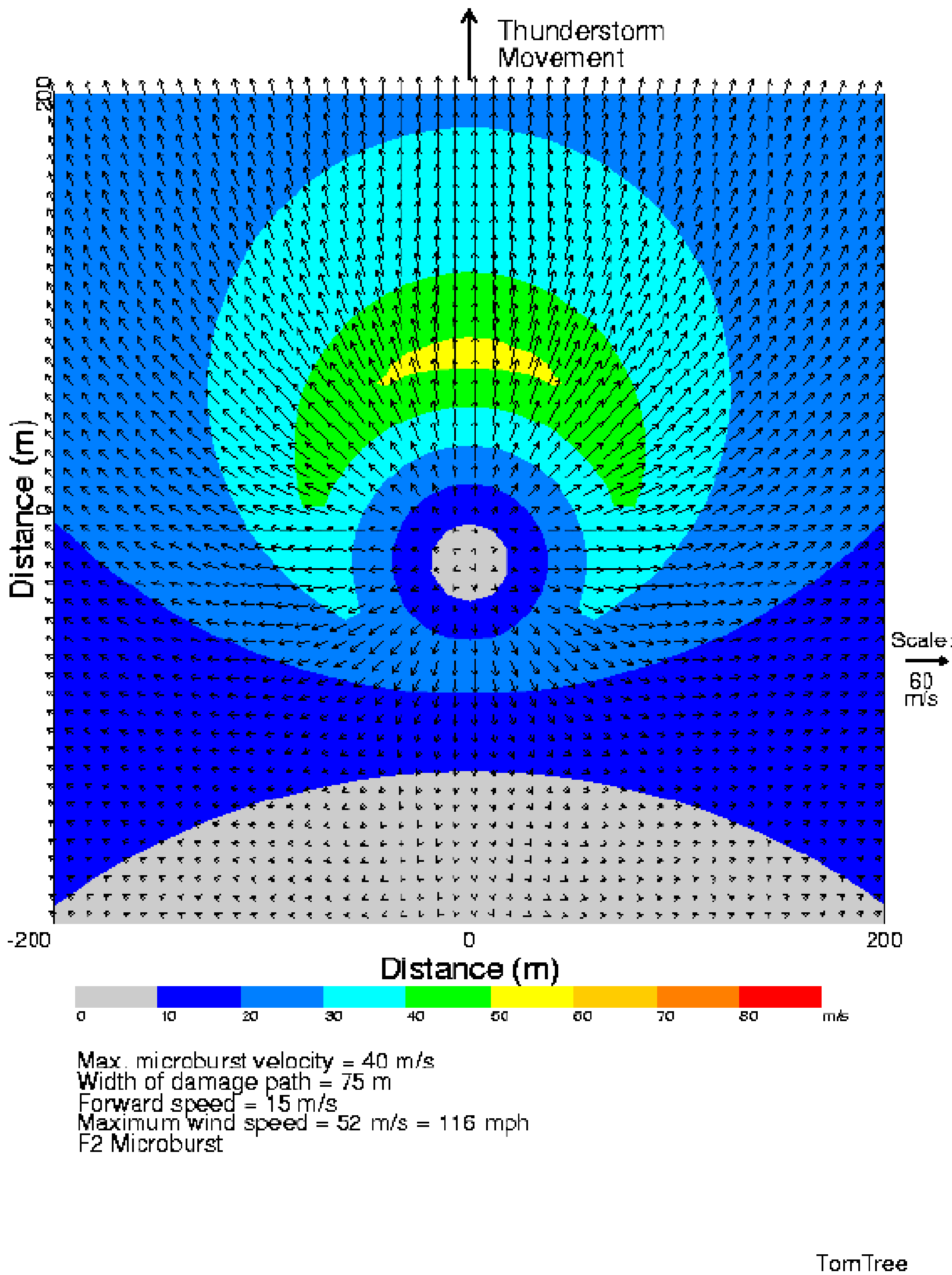


Figure 3.3.13a: Same as Fig. 3.3.12a, with a higher maximum microburst velocity.

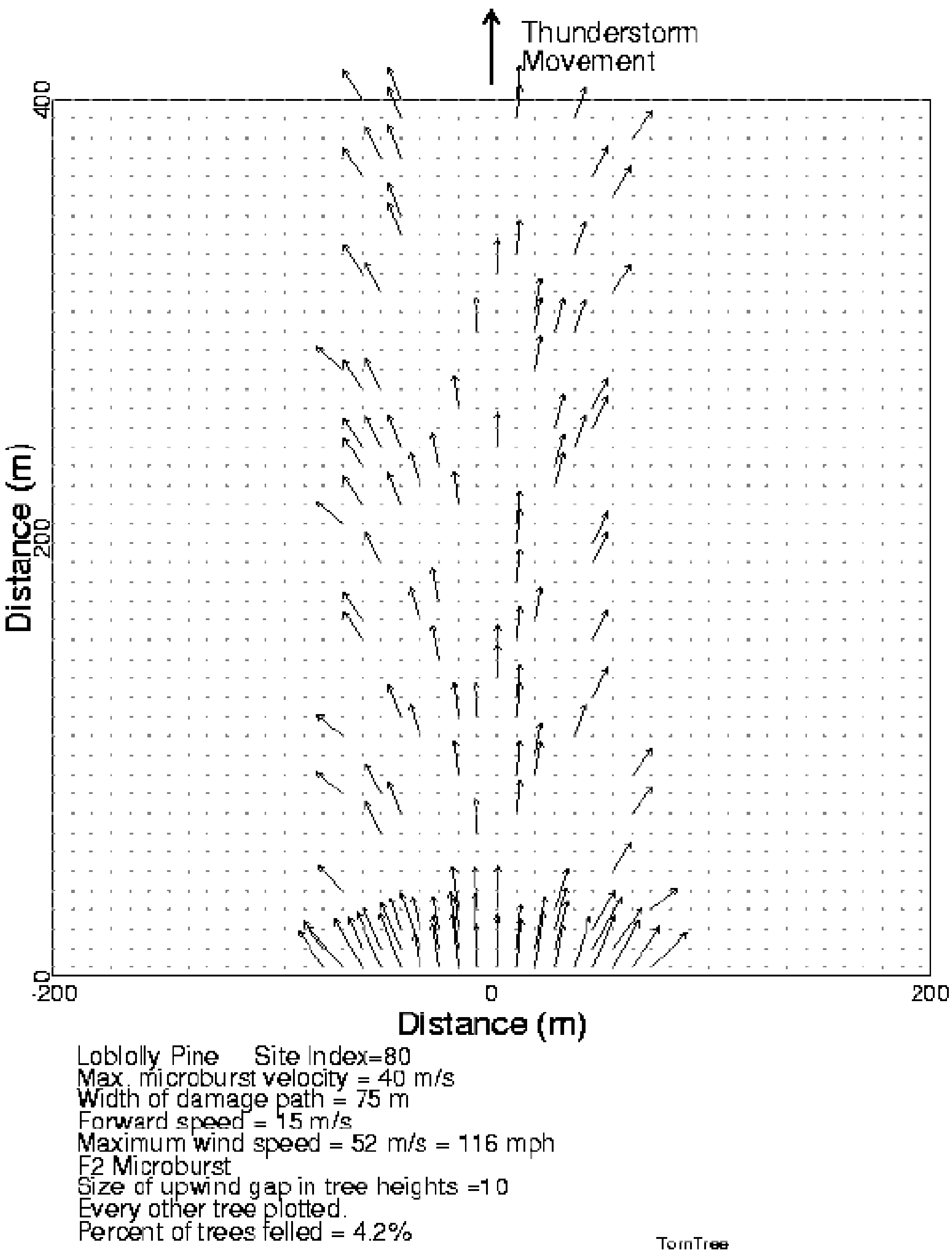


Figure 3.3.13b: Tree fall pattern for microburst in Fig. 3.3.13a.

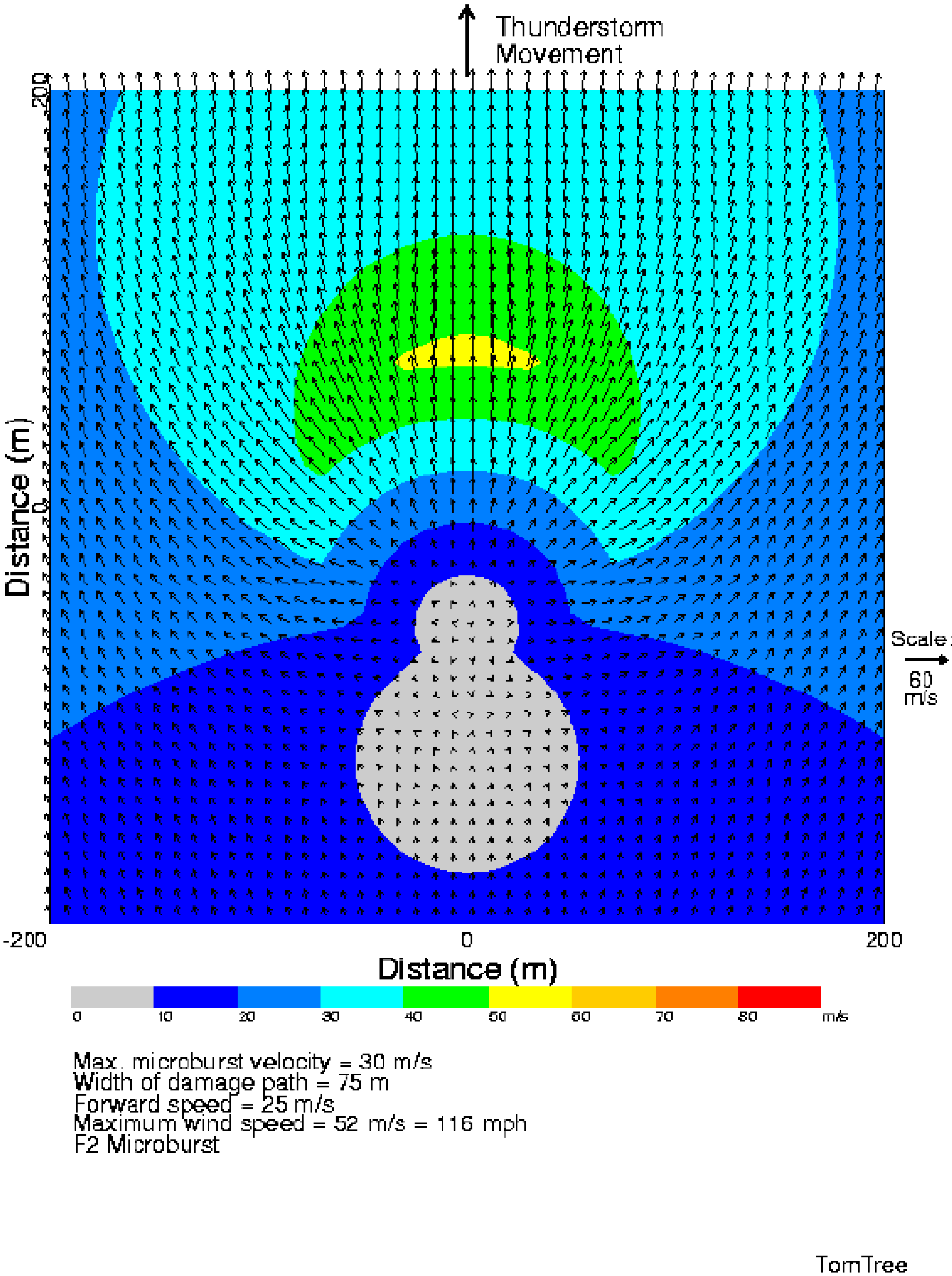


Figure 3.3.14a: Same as Fig. 3.3.12a, with a higher forward speed.

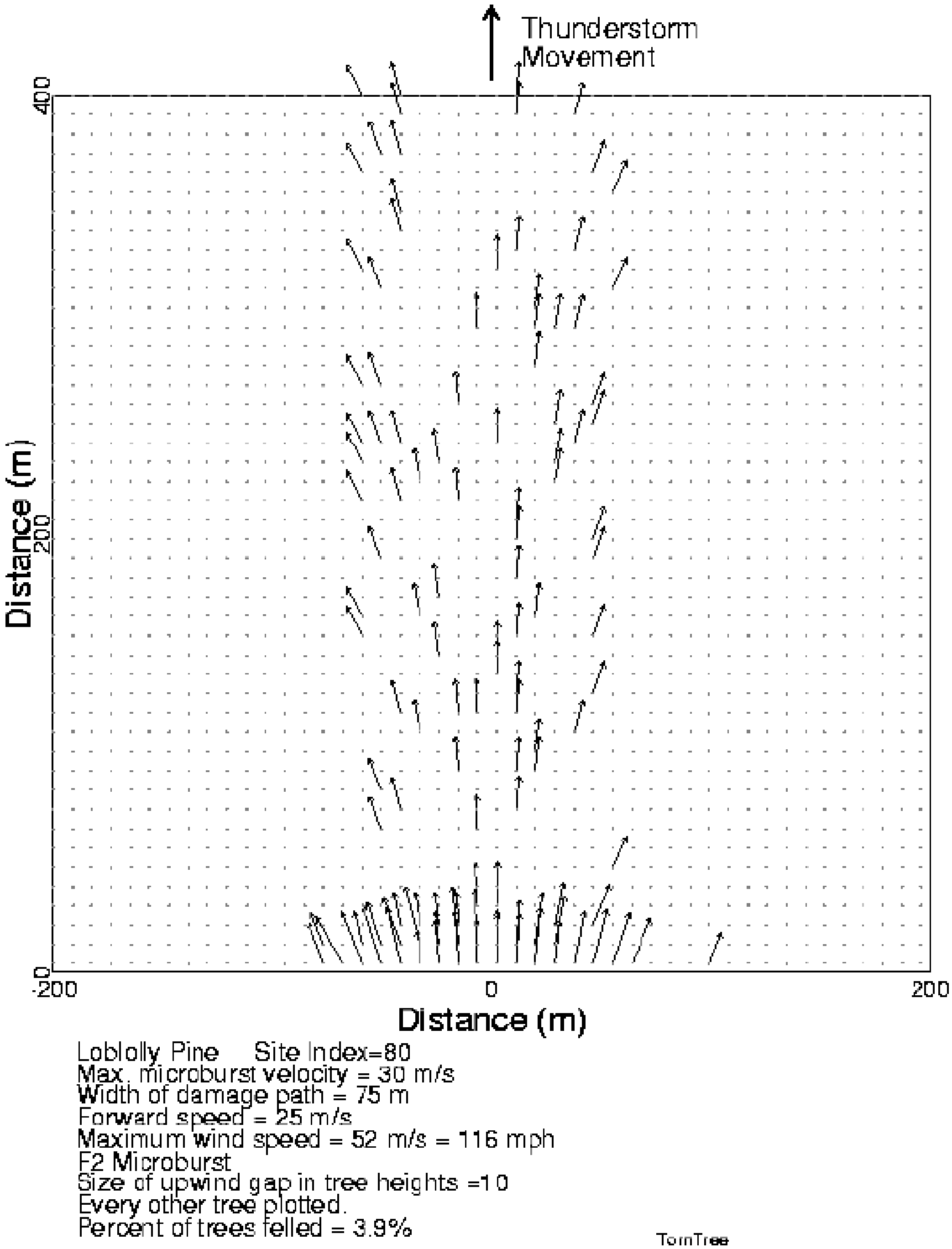


Figure 3.3.14b: Tree fall pattern for microburst in Fig. 3.3.14a.

4. Summary and Future Work

4.1 Summary

A model (TornTree) has been created to estimate near-surface tornado wind speeds from tree damage patterns. The model is composed of a vortex model and a tree model. The vortex model is based on the combined Rankine vortex with forward motion and inflow added to it. The tree model is based on research done by Peltola et al. (1999). It determines the critical turning moment of a tree and compares it to the turning moment caused by the vortex wind speeds. If the critical turning moment is exceeded the tree's stem will break and the tree will fall.

The model assumes flat terrain and a steady-state vortex. There is no tree interaction and the trees are evenly spaced in the stand. The model outputs graphical files of the vortex and the tree fall pattern, as well as the critical wind speeds of the trees.

The model was compared with the HWIND model (Peltola et al. 1999), which determined the critical wind speed of Scots pine. The TornTree model's critical wind speeds for Scots pine agreed well with those from the HWIND model.

Due to the prevalence of loblolly pine in the Southeastern United States, the model was also run for loblolly pine. The critical wind speeds for stem breakage of loblolly pine were greater than those for Scots pine. This is expected since the modulus of rupture is so much greater for loblolly pine.

The TornTree model was created to provide a fast and easy way to estimate near-surface tornado wind speeds from tree damage patterns; and to assist in damage assessment teams analysis of tornado damage.

4.2 Future Work

This model is a work in progress and thus leaves room for improvement. At this stage the model assumes flat terrain. Differing terrain could be added to the model in the future. Tree interactions are also very important in creating tree fall patterns. More species of trees could be added and many species could be mixed together to model actual forests.

The Rankine vortex is a very simple, analytical vortex model that is two-dimensional and steady state. A numerical simulation could be used in place of the Rankine vortex in the model to give more detailed, three-dimensional, non-steady state vortex velocities.

Modeling air flow through a forest is no easy task. This model assumes a logarithmic profile and uses empirical relationships derived from wind-tunnel experiment to modify the profile when the wind enters the forest. Perhaps the model should be run with the exponential wind profile from Oliver and Mayhead (1974) for crops. This profile would work well in the stand, but the logarithmic profile would have to be used at the stand edge, and as yet there is no way to move from the logarithmic profile to the exponential profile as the wind enters the stand.

At this stage there are two ways to compare damage patterns from the model with actual ones: optical comparison and percent of trees down. While these do an adequate job of estimating maximum wind speed and F-number, they are very subjective and are inadequate at estimating tangential and radial velocities. A better way to compare would be to have a program break down the photograph of actual damage into cells. This program would then read in the number or percent of trees down along with the angle of felling in that cell and compare it to the same cell from the model damage plot. If number or percent and the angle were within some specified threshold for all or a specified portion of the cells it would be deemed a good estimation. Since the model plots are from directly overhead it would be difficult to compare them using this method if the photograph is taken from a different point of view. However, if the location of the stand and the direction of tornado motion are inputted into GIS software it will output the direction of tree fall from an overhead perspective.

One final improvement would be to make the model run in PC and Macintosh environments so it could be more accessible.

References

Baldwin, V. C., Jr., 1987: Green and Dry-Weight Equations for Above-Ground components of Planted Loblolly Pine Trees in the West Gulf Region. *South. J. Appl. For.*, **11**, 212-218.

Clark, A. III, 1991: Weight and Volume Equations for Loblolly Pine in Eastern North Carolina. Final Report of a Cooperative Study Between USDA Forest Service Southeastern Forest Experiment Station and Federal Paper Board Co., Inc.

Clark, A. III, R. A. Souter and B. E. Schlaegel,1991: Stem Profile Equations for Southern Tree Species. Res. Pap. SE-282. Asheville, NC: USDA, Forest Service, Southeastern Forest Experiment Station. 124pp.

Franklin, E. C., 2000: Personal communication.

Fredericksen, T. S., R. L. Hedden, S. A. Williams, 1993: Testing loblolly pine wind firmness with simulated wind stress. *Can. J. For. Res.*, **23**, 1760-1765.

Fujita, T. T., 1978: Manual of downburst identification for Project NIMROD. *SMRP Res. Paper No. 210*, Univ. of Chicago, 82pp.

Fujita, T. T., 1981: Tornadoes and downbursts in the context of generalized planetary scales. *J. Atmos. Sci.*, **38**, 1511-1534.

Gardiner, B. A. and G. R. Stacey, 1995: Designing forest edges to improve wind stability. *Scott. For. Comm. Tech. Pap. No. 16,* 8pp.

Gardiner, B. A., G. R. Stacey, R. E. Belcher and C. J. Wood, 1997: Field and wind tunnel assessments of the implications of respacing and thinning for tree stability. *Forestry*, **70**, 233-252.

Grace, J., 1977: Plant response to wind. Academic Press, New York, 204pp.

Hakkila, P., 1971: Coniferous branches as a raw material source. A sub-project of the joint Nordic research programme for the utilization of logging residues. *Commun. Inst. For. Fenn.*, **No. 75**, 60pp.

Hamilton, Rick. Woodland Owner Notes: Forest Soils and Site Index. North Carolina Cooperative Extension Service. 1993.

Hedden, R. L., T. S. Fredericksen and S. A. Williams, 1995: Modeling the effect of crown shedding and streamlining on the survival of loblolly pine exposed to acute wind. *Can. J. For. Res.,* **25**, 704-712.

Jones, H. G., 1983: Plants and microclimate. A quantitative approach to environmental plant physiology. Cambridge University. Press, Cambridge, U.K.

Laasasenaho, J., 1982: Taper curve and volume functions for pine, spruce and birch. *Commun. Inst. For. Fenn.,* **No. 108**, 74pp.

Letzmann, J., 1923: Das Bewegungsfeld in Fuss einer fortschreitenden Wind- oder Wasserhose. *Aeta Comm. Univer. Dorpat,* **AIII**, 1-136.

Lewellen, W. S., 1993: Tornado Vortex Theory. *The Tornado: Its Structure, Dynamics, Prediction and Hazards*, C. Church, D. Burgess, C. Doswell and R. Davies-Jones, Eds. Geophysical Monograph 79, American Geophysical Union, Washington, D.C., 19-39.

Mattheck, C. and H. Breloer, 1994: The body language of trees: A handbook for failure analysis, D. Lonsdale, Ed. HMSO, London, 240pp.

Monteith, J. H., 1975: Principles of environmental physics. 1st Ed. Edward Arnold, London.

Morgan, J. and M. G. R. Cannell, 1994: Shape of tree stems: A reexamination of the uniform stress hypothesis. *Tree Physiol.*, **5**, 63-74.

Oliver, H. R. and G. J. Mayhead, 1974: Wind Measurements in a Pine Forest During a Destructive Gale. *Forestry*, **47**, 185-194.

Peltola, H., 1996: Model computations on wind flow and turning moment by wind for Scots pine along the margins of clear-cut areas. *For. Ecol. Manage.*, **83**, 203-215.

Peltola, H., 2000: Personal communication.

Peltola, H. and S. Kellomäki, 1993: A mechanistic model for calculating windthrow and stem breakage of Scots pine at stand edge. *Silva Fenn.*, **27**, 99-111.

Peltola, H., S. Kellomäki, A. Hassinen, M. Lemettinen and J. Aho, 1993: Swaying of trees as caused by wind: analysis of field measurements. *Silva Fenn.*, **27**, 113-126.

Peltola, H., M-L. Nykänen and S. Kellomäki, 1997: Model computations on the critical combination of snow loading and windspeed for snow damage of Scots pine, Norway spruce and Birch sp. at stand edge. *For. Ecol. Manage.*, **95**, 229-241.

Peltola, H., S. Kellomäki, H. Välsänen and V. –P. Ikonen, 1999: A mechanistic model for assessing the risk of wind and snow damage to single trees and stands of Scots pine, Norway spruce and birch. *Can. J. For. Res.*, **29**, 647-661.

Pennala, E., 1980: Lujuusopin perusteet. **No. 407**, Otakustantamo, Espoo, Finland. [In Finnish].

Peterson, R. E., 1992: Johannes Letzmann: A Pioneer in the Study of Tornadoes. *Wea. Forecasting*, **7**, 166-184.

Petty, J. A., C. Swain, 1985: Factors influencing stem breakage of conifers in high winds. *Forestry*, **58**, 75-85.

Rankine, W. J., A Manual of Applied Mechanics, Charles Griffin, London, 1882.

Roth, J. T., 1990: Features of the Raleigh tornadic storm based on analysis of damage. Masters Thesis. North Carolina State University, Raleigh, NC. 112pp.

Trapp, R. J. and B. H. Fiedler, 1993: Numerical Simulation of Tornadolike Vortices in Asymmetric Flow. *The Tornado: Its Structure, Dynamics, Prediction and Hazards*, C. Church, D. Burgess, C. Doswell and R. Davies-Jones, Eds. Geophysical Monograph 79, American Geophysical Union, Washington, D.C., 49-54.

Turner, J. S., 1966: The constraints imposed on tornado-like vortices by the top and bottom boundary conditions. *J. Fluid Mech.,* **25**, 377-400.

Vandersip, C., 1998: A Single-Doppler Radar Study of Kinematic and Structural Characteristics of Mesocyclones in the Southeastern and Great Plains Regions of the United States. Masters Thesis. North Carolina State University, Raleigh, NC. 166pp.

Wahlenberg, W. G., Loblolly Pine: Its Use, Ecology, Regeneration, Protection, Growth and Management. The School of Forestry, Duke University, Durham, NC, 1960.

Ward, N. B., 1972: The Exploration of Certain Features of Tornado Dynamics Using a Laboratory Model. *J. Atmos. Sci.,* **29**, 1194-1204.

Wood, C. J., 1995: Understanding wind forces on trees. Wind and Trees, M. P. Coutts and J. Grace Eds. Cambridge Univ. Press, New York, NY, 133-164.

Zrnic, D., D. W. Burgess and L. Hennington, 1985: Doppler Spectra and Estimated Windspeed of a Violent Tornado. *J. Clim. Appl. Met.,* **24**, 1068-1081.

Appendix A
Modeling of Loblolly Pine

1. Crown area distribution

To determine the cross-sectional area of each one-meter segment in the crown, the crown is approximated by two triangles (Franklin 2000). These triangles have a common base that is twice the length of the longest branch, which is also approximately the spacing between the trees. The upper triangle is the live crown and is assumed to be 30% of the tree height. The lower triangle is the dead crown and is assumed to be 10% of the tree height.

Example:

To start, you know that the base of the triangle is 4.9m and that the height of the live crown is 8m. To find the base of the triangle that is 7m tall use the following relationship:

$$\frac{totaltriangleheight}{totaltriangleheight\ ?\ xm}\ ?\ \frac{branchlength(totaltriangle)}{branchlength(totaltriangle\ ?\ xm)}$$

and solve for the unknown branch length (branchlength(totaltriangle – 1m), where x is the height of the new triangle (Peltola 2000). For the following example:

$$\frac{8m}{7m}\ ?\ \frac{4.9m}{branchlength(totaltriangle\ ?\ 1m)},$$

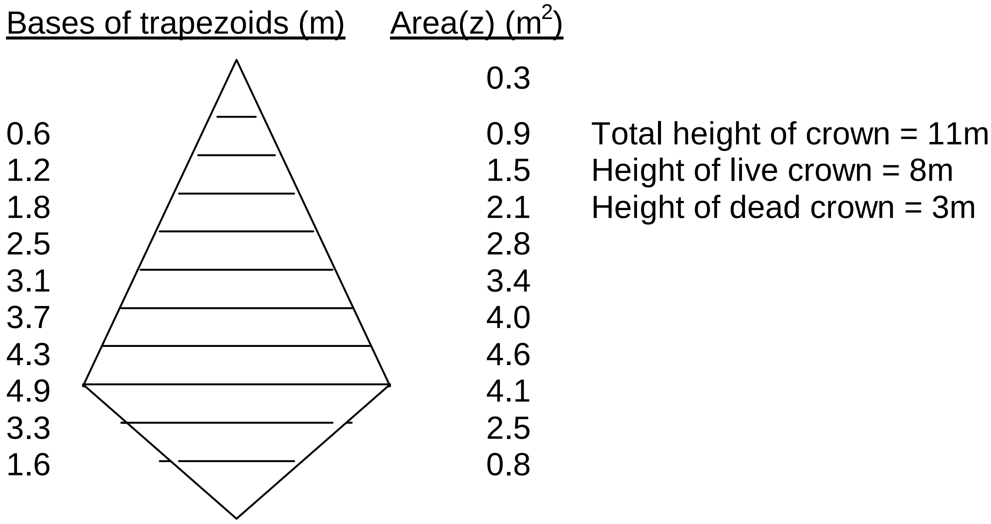
and the new branch length is 4.3m. Once one has obtained the branch lengths, the area of each trapezoid can be calculated from:

$$area(z)\ ?\ 0.5\ ?\ height\ ?\ (base1\ ?\ base2).$$

For the trapezoid from above:

$$area(z)\ ?\ 0.5\ ?\ 1m\ ?\ (4.9m\ ?\ 4.3m),$$

therefore, area(z)=4.6m^2. The total crown area is the sum of the areas of the one-meter segments and is 27 m^2 for this example.



2. Stem Area Distribution

To calculate stem area distribution, the stem is divided into one-meter segments. The diameter of the stem is determined at the base of the tree and at one-meter intervals to the top of the tree. The stem diameter of a loblolly pine can be calculated at any height if the height of the tree and DBH are known, using the following method from Clark et al. (1991).

First one calculates the diameter at a height of 17.3 feet (F) using the following equation:

$$F \ ? \ D(a \ ? \ b(17.3/H)^2)$$

Where D is the DBH, H is the tree height and a and b are coefficients determined by tree species.

Next, several indicator variables are determined:

I_S=1 if h ? 4.5 ft.

=0 otherwise

I_B=1 if 4.5 ? h ? 17.3

=0 otherwise

I_T=1 if h ?17.3

=0 otherwise

I_M=1 if h ? (17.3+a(H-17.3))

=0 otherwise

Finally, the diameter at the top of each one-meter segment (d) can be calculated from:

$$d \; ? \; [I_S\{D^2(1 \,?\, (c \,?\, e/D^3)((1 \,?\, h/H)^r \;?\; (1 \,?\, 4.5/H)^r)/(1 \,?\, 4.5/H)^r))\}$$
$$? \; I_B\{D^2 \;?\; (D^2 \;?\; F^2)((1 \,?\, 4.5/H)^p \;?\; (1 \,?\, h/H)^p)/((1 \,?\, 4.5/H)^p \;?\; (1 \,?\, 17.3/H)^p)\}$$
$$? \; I_T\{F^2(b(((h \,?\, 17.3)/(H \;?\; 17.3)) \,?\, 1)^2 \;?$$
$$I_M((1 \,?\, b)/a^2)(a \,?\, (h \,?\, 17.3)/(H \;?\; 17.3))^2)\}]^{0.5},$$

where a,b,c,e,r and p are coefficients based on tree species.

The area of each one-meter segment is now calculated using the area of a trapezoid, with the diameters as the bases and a height of one meter.

3. Crown Mass Distribution

In order to calculate the crown mass distribution the total crown mass must be known. Baldwin (1987) has developed prediction equations for crown mass based on 130 sample loblolly pine trees in southern Louisiana. The following regression equation can be used to calculate the dry or green weight of crown components.

$$\ln(W) \; ? \; b_1' \; ? \; b_2 \ln(D) \; ? \; b_3 \ln(H),$$

where W is the predicted weight (lb.) of crown component, D is the stem diameter at breast height (in.), H is the total tree height (ft.) and b_1', b_2 and b_3 are coefficients estimated from the data. Using the above equation the green weights of wood, bark and foliage in the crown are calculated and summed to estimate the total crown mass. For the example tree in Section 1 the total mass is 135.8 kg. To obtain the mass of each one-meter crown segment, the following relationship is used (Peltola 2000),

$$\frac{area(z)}{totalarea} \; ? \; \frac{mass(z)}{totalmass}.$$

4. Stem Mass Distribution

To calculate the stem mass distribution the total stem mass must be known. This can be calculated for loblolly pine using the following equation from Clark (1991):

$$TOTSTEM ? 0.17567(DBH^2)^{1.00751}(THT)^{0.95925}$$

where TOTSTEM is the total stem weight and THT is the total tree height. The mass for each one-meter stem segment can be calculated in the same manner as the crown mass distribution.

Table A.1: Loblolly pine characteristics (from Wahlenberg 1960).

Site Index	Age (yr.)	Height (m)	DBH (cm)	Spacing (m)	Taper
60	20	11.6	13.5	2.5	85.9
	30	14.9	18	3.1	82.8
	40	17.1	20.8	3.5	82.2
	50	18.3	22.9	3.7	79.9
	60	19.2	24.1	3.9	79.7
	70	19.8	25.1	4	78.9
	80	20.4	25.9	4.1	78.8
70	20	13.7	15.5	2.8	88.4
	30	17.3	20.8	3.5	83.2
	40	19.8	24.1	3.9	82.2
	50	21.3	26.2	4.2	81.3
	60	22.6	27.9	4.4	81
	70	23.2	29	4.5	80
	80	23.8	29.7	4.6	80.1
80	20	15.5	17.5	3.1	88.6
	30	20.1	23.6	3.8	85.2
	40	22.6	27.2	4.3	83.1
	50	24.4	29.7	4.6	82.2
	60	25.6	31.5	4.8	81.3
	70	26.5	32.8	5	80.8
	80	27.4	33.5	5.1	81.8
90	20	17.4	19.6	3.4	88.8
	30	22.6	26.4	4.2	85.6
	40	25.6	30.5	4.6	83.9
	50	27.4	33.3	5	82.3
	60	29	35.3	5.2	82.2
	70	29.9	36.8	5.4	81.2
	80	30.8	37.8	5.5	81.5
100	20	19.5	21.6	3.6	90.3
	30	25	29.2	4.4	85.6
	40	28.3	34	5	83.2
	50	30.5	37.1	5.3	82.2
	60	32	39.1	5.6	81.8
	70	33.2	40.6	5.8	81.8
	80	34.1	41.9	5.9	81.4
110	20	21.3	23.9	3.8	89.1
	30	27.4	32.3	4.7	84.8
	40	31.1	36.6	5.2	85
	50	33.5	40.6	5.6	82.5
	60	35.4	43.2	5.9	81.9
	70	36.6	44.7	6.1	81.9
	80	37.5	46	6.3	81.5
120	20	23.1	25.7	4	89.9
	30	29.9	35.6	4.9	84
	40	33.5	40.9	5.5	81.9
	50	36.6	44.5	5.9	82.2
	60	38.4	47	6.2	81.7
	70	39.9	49	6.4	81.4
	80	40.8	50.3	6.6	81.1

Appendix B
Model Walk Through

1. **Specifications**

The TornTree model is a Fortran77 program compiled on a Sun Ultra 10. The model takes from two to three minutes to run.

2. **Walk Through**

To run the model type “torntreev1” at the prompt. A brief summary will appear and the user will be prompted to hit <enter>. Now the user is asked what species of trees he is interested in. Next the user chooses either the Site Index for loblolly or the age class for Scots pine. There is an option to input your own tree data for loblolly pine. If loblolly is chosen and then a Site Index, the user must input a random number seed in order to randomly distribute different aged trees throughout the stand. If Scots pine is chosen, there is an option run the model on a random distribution from all the age classes.

The next option is to run the model for a tornado or a microburst. If tornado is chosen the user must make a best guess of the maximum tangential velocity of the tornado. The model then asks for the radius of maximum winds. This can be approximated by the width of the damage path. The model will now ask for the maximum radial wind speed. This is the inflow wind speed. Next the model asks for the forward speed of the tornado. Now the user is prompted to enter the upwind gap size. This is the distance in tree heights of open area upwind of the forest. Finally, the model asks the user if they would like to create a color contour plot of the vortex.

If the user chooses to run the model for a microburst he will be asked for a guess of the maximum microburst velocity, the width of the damage path and the forward speed of the thunderstorm.

Once all the inputs have been obtained the model will run and upon completion the user can print out the output files of the vortex, the contour plot, the tree fall pattern and the critical wind speeds (if applicable).

3. Screen Dump

Loblolly/Tornado run

```
*******************************
 ********************************
   ****************************
   **************************
  **************************
 ************************
 ********************
  ****************
   *************
     ********
       ******
        *****
         ****
          ****
           ****
            ***
             **

 TTTTTTT                   TTTTTTT
    T    oo    rrrrr  n nn    T   rrrrr   eeeee     eeeee
    T   o  o   r      n   n   T   r      e     e   e     e
    T  o    o  r      n   n   T   r      eeeeee    eeeeee
    T   o  o   r      n   n   T   r       e         e
    T    oo    r      n   n   T   r         eeee      eeee

 **********************************************************
 *   TornTree - A model to estimate tornado windspeeds    *
 *              from tree damage patterns.                *
 *                                                        *
 *   Created by:  Andrew P. Holland - holland6@yahoo.com *
 *                                                        *
 *    The TornTree model is used to estimate tornado      *
 * windspeeds from tree damage patterns.  It is the       *
 * combination of a vortex model (Peterson 1992) and a    *
 * mechanistic tree model designed by Peltola et al.      *
 * (1999). The model calculates the windspeed and turning*
 * moment at each tree.  If this turning moment is        *
 * greater than the resistance to stem breakage for       *
 * the tree then it is broken.  An output file is         *
 * created showing the orientation of the fallen trees.   *
 *                                                        *
 * Peltola et al., 1999. Canadian J. of Forest Research, *
 * 29, 647-661.                                           *
 *                                                        *
 * Peterson, R.E., 1992.  Weather and Forecasting, 7,     *
 * 166-184.                                               *
```

```
 *                                                        *
 **********************************************************
Type <return> to continue.



 ****************************************************
 *                                                  *
 *(1)Run the model for loblolly pine.               *
 *(2)Run the model for Scots pine.                  *
 *                                                  *
 ****************************************************

Enter an option (1-2) ->1

 ****************************************************
 *                                                  *
 * Site index is the height in feet of 50 year old  *
 * trees in this stand and is based on soil         *
 * properties.                                      *
 *                                                  *
 *(1)Site Index = 60                                *
 *(2)Site Index = 70                                *
 *(3)Site Index = 80                                *
 *(4)Site Index = 90                                *
 *(5)Site Index = 100                               *
 *(6)Site Index = 110                               *
 *(7)Site Index = 120                               *
 *(8)Input data for any loblolly pine tree.         *
 *                                                  *
 ****************************************************

Enter an option (1-8) ->1
 Enter a seed for the random number generator.
 *This number can be any number of digits.
 *This allows the trees to be randomly
 *distributed by age.
7584
 ****************************************************
 *                                                  *
 *      Choose one:                                 *
 *      (1) Tornado                                 *
 *      (2) Microburst                              *
 *                                                  *
 ****************************************************

Enter an option (1-2) ->1


What is the maximum wind speed in m/s?
 *This is your initial guess.*
45
 What is the radius of max winds in meters?
75
 What is the maximum radial wind speed in m/s?
20
```

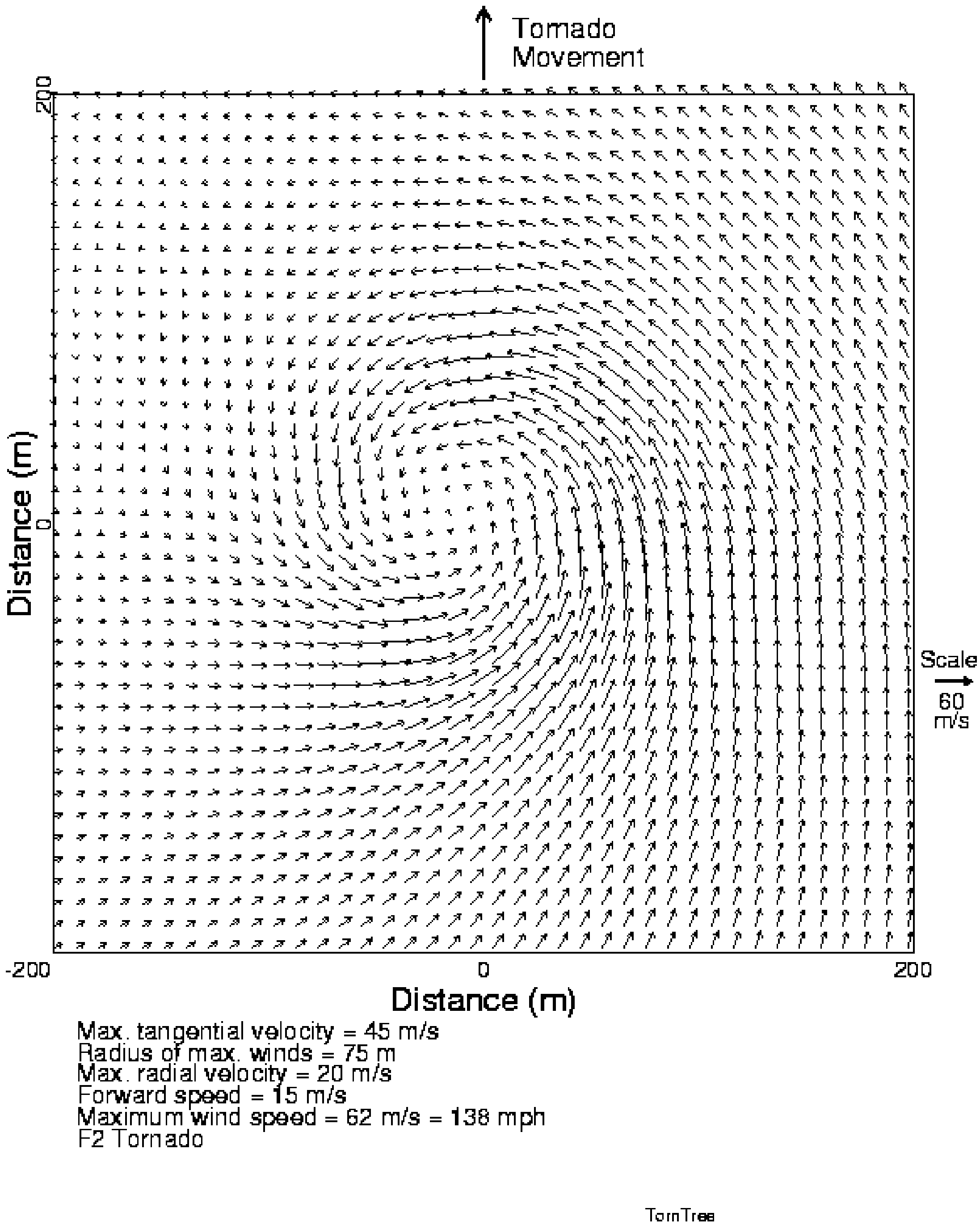
```
 What is the forward speed of the tornado in m/s?
15
 What is the size of the upwind gap in tree heights?
 * 0<gapsize<=10 *
10
 Would you like to print out a color contour plot
 of the vortex winds (y/n)?
y


Printing plot of vortex winds. . . .(tornado.ps)
Printing contour plot . . . . .(contour.ps)
Printing plot of fallen trees . . . .(trees.ps)
```

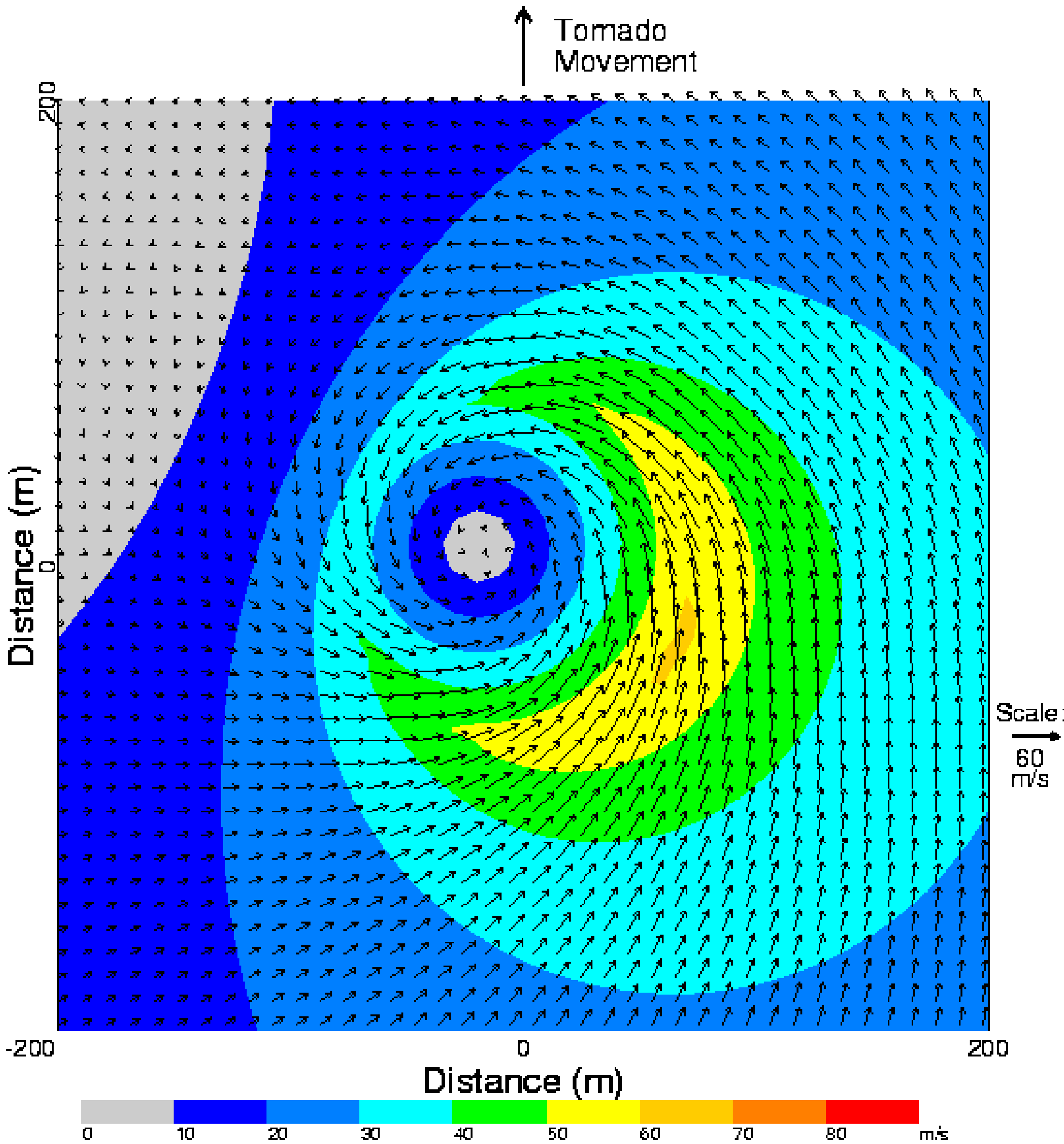
Appendix C

Sample Files from the Model

1.tornado.ps



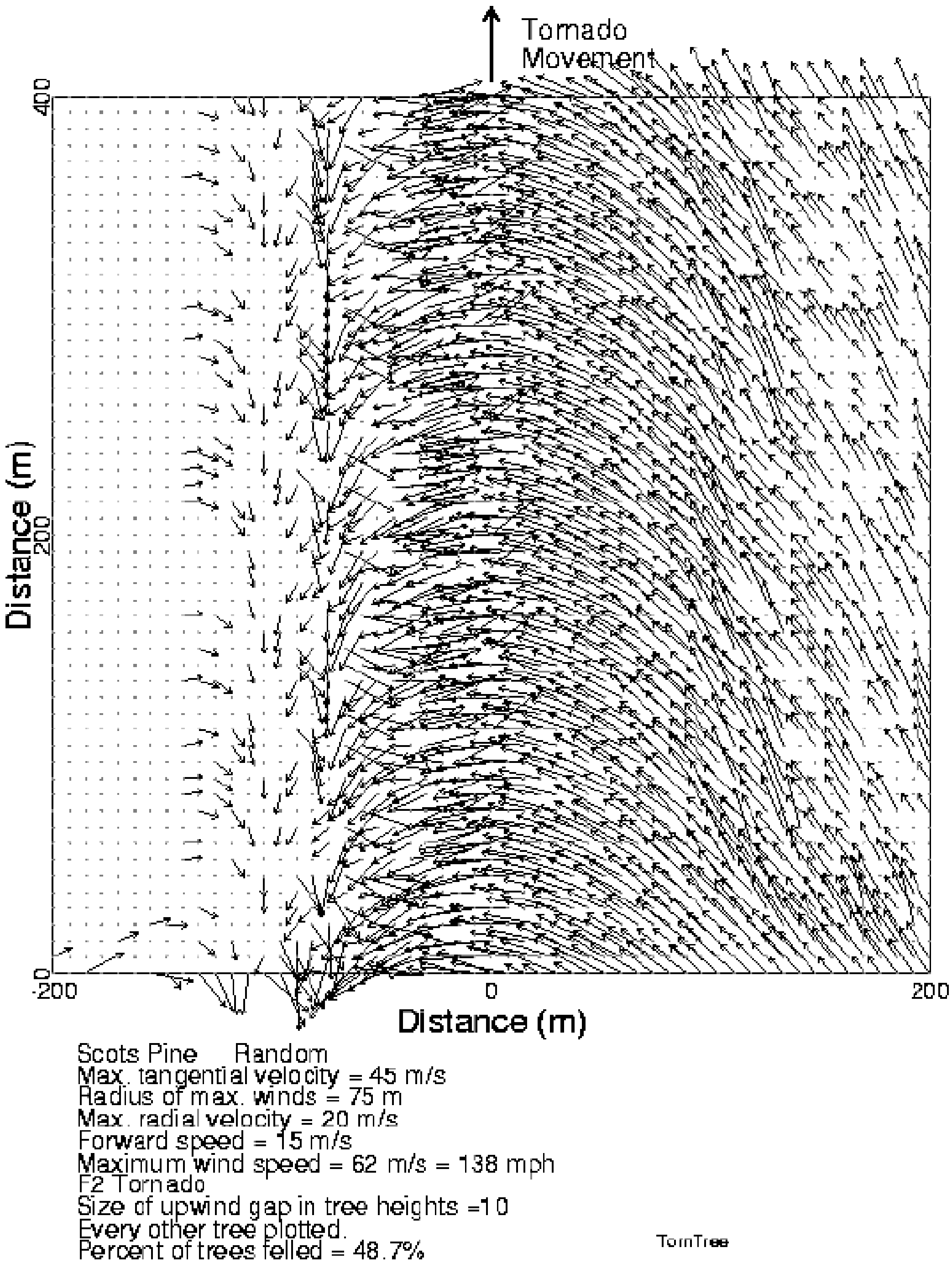
2. contour.ps



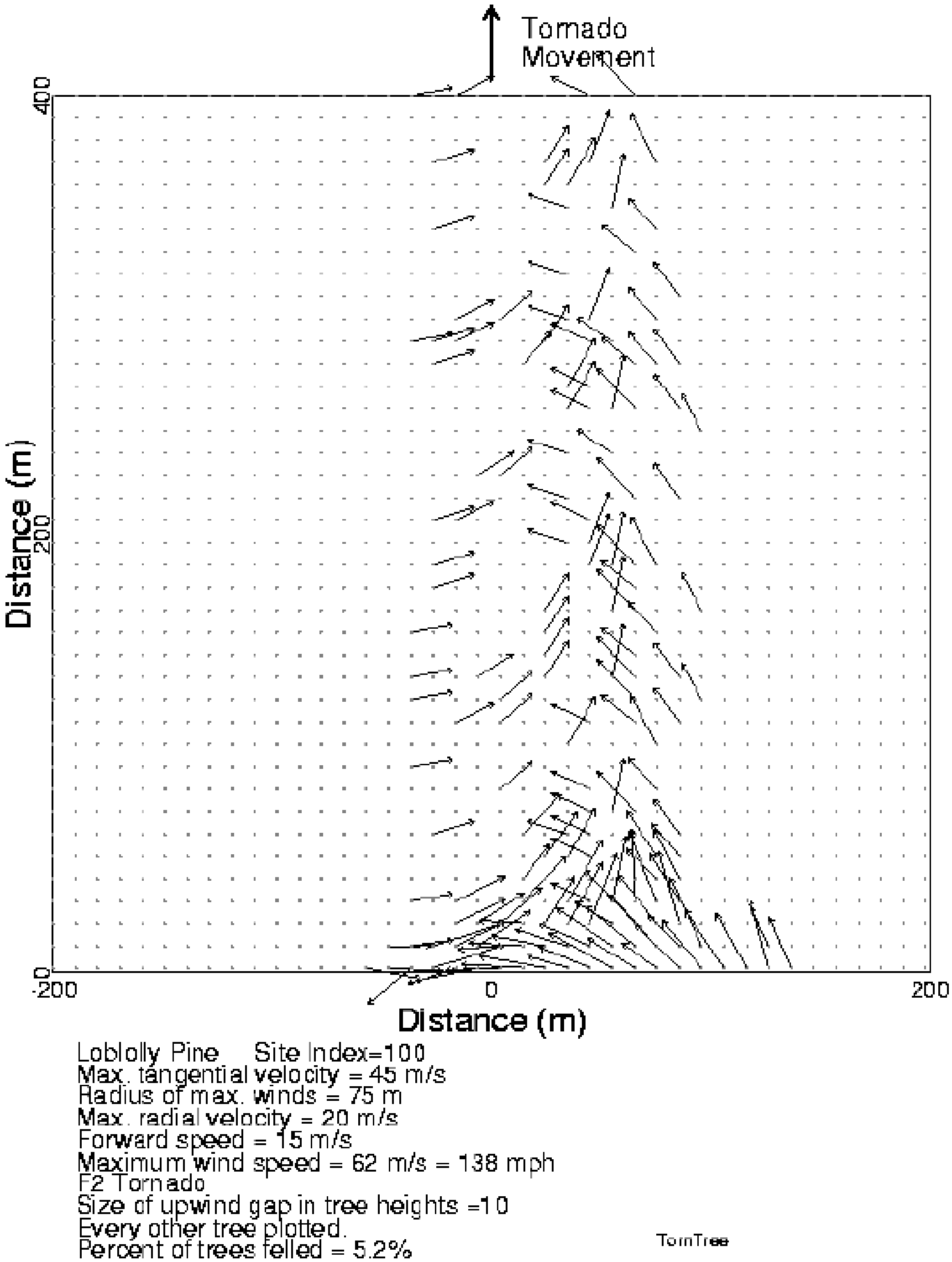
Max. tangential velocity = 45 m/s
Radius of max. winds = 75 m
Max. radial velocity = 20 m/s
Forward speed = 15 m/s
Maximum wind speed = 62 m/s = 138 mph
F2 Tornado

TornTree

3. trees.ps (Scots pine)



4. trees.ps (loblolly pine)



5. critwind.dat (Scots pine)

Scots pine
height= 9.3m
diameter at breast height= 9.0cm

```
tree spacing=  1.7m
tree taper= 103.3
upwind gapsize= 10.0 tree heights

distance from edge= 0h    critical wind speed=   15.9 m/s
distance from edge= 1h    critical wind speed=   26.5 m/s
distance from edge= 2h    critical wind speed=   29.9 m/s
distance from edge= 3h    critical wind speed=   30.9 m/s
distance from edge= 4h    critical wind speed=   31.2 m/s
distance from edge= 5h    critical wind speed=   31.2 m/s
distance from edge= 6h    critical wind speed=   31.2 m/s
distance from edge= 7h    critical wind speed=   31.2 m/s
distance from edge= 8h    critical wind speed=   31.2 m/s
distance from edge= 9h    critical wind speed=   31.2 m/s
```

6. critwind.dat (loblolly pine)

```
loblolly pine
height= 27.4m
diameter at breast height= 33.3cm
tree spacing=  5.0m
tree taper=  82.3
upwind gapsize= 10.0 tree heights

distance from edge= 0h    critical wind speed=   41.0 m/s
distance from edge= 1h    critical wind speed=   56.9 m/s
distance from edge= 2h    critical wind speed=   64.3 m/s
distance from edge= 3h    critical wind speed=   66.5 m/s
distance from edge= 4h    critical wind speed=   67.1 m/s
distance from edge= 5h    critical wind speed=   67.2 m/s
distance from edge= 6h    critical wind speed=   67.3 m/s
distance from edge= 7h    critical wind speed=   67.3 m/s
distance from edge= 8h    critical wind speed=   67.3 m/s
distance from edge= 9h    critical wind speed=   67.3 m/s
```

Appendix D

TornTree Version 1.0

```
        program torntreev1

************************************************************************     *
                                                                        *
*   TornTree - A model to estimate tornado windspeeds from tree damage *
*               patterns.                                              *
*                                                                       *
*   Version: 1.0                                                       *
*                                                                       *
*                                                                      *
*                                                                       *
*   Created by:  Andrew P. Holland                                     *
*                                                                       *
*   Advisory committe:  Dr. Allen Riordan                              *
*                       Dr. Carlyle Franklin                           *
*                       Dr. Jerry Davis                                *
*                                                                      *
*                                                                       *
*       The TornTree model is used to estimate tornado windspeeds from *
*       tree damage patterns.  It is the combination of a vortex model *
*       and a mechanistic tree model.  The vortex model is a combined  *
*       Rankine vortex with forward motion (Peterson 1992). The tree   *     *
model is based on work by Peltola et al. (1999).  The model    *     *
calculates the windspeed at each point in the grid and         *     *
calculates the turning moment at each grid point.  Each grid   *     *
point, or tree, has a resistance to stem breakage calculated   *     *    for
it.  If the turning moment exceeds this resistance, the    *     *         tree
is considered to be downed.  The model outputs a graphical*     *    file of
the orientation of the fallen trees.  This output can  *     *      then be
compared with pictures of actual damage to estimate the*      *     windspeed
and forward speed of the tornado.                     *
*                                                                      *
*       Peltola H., S. Kellomaki, H. Vaisanen, and V.-P. Ikonen, 1999: *
*       A mechanistic model for assessing the risk of wind and snow    *
*       damage to single trees and stands of Scots pine, Norway spruce,*
*       and birch.  Canadian Journal of Forest Research, 29, 647-661.  *
*                                                                       *
*       Peterson, R.E., 1992:  Johannes Letzmann: A Pioneer In the     *
*       Study of Tornadoes.  Weather and Forecasting, 7, 166-184.      *
*                                                                      *
*       Outputs:  "tornado.ps"  -> plot of vortex wind field           *
*                 "contour.ps"  -> color contour plot of vortex wind   *     *
                                    field                              *
*                 "trees.ps"    -> plot of downed trees                *
*                 "critwind.dat -> critical wind speeds                *
*                                                                       *
************************************************************************

      call clrscrn

      write(*,*)"******************************"
```

```
      write(*,*)"********************************"
      write(*,*)"  *****************************"
      write(*,*)"  ***************************"
      write(*,*)" **************************"
      write(*,*)"************************"
      write(*,*)"********************"
      write(*,*)" ****************"
      write(*,*)"  *************"
      write(*,*)"    ********"
      write(*,*)"      ******"
      write(*,*)"       *****"
      write(*,*)"        ****"
      write(*,*)"         ****"
      write(*,*)"          ****"
      write(*,*)"           ***"
      write(*,*)"            **"
      write(*,*)
      write(*,*)"TTTTTTT                    TTTTTTT"
      write(*,*)"   T    oo    rrrrr  n nn    T   rrrrr    eeeee     eeeee”
      write(*,*)"   T   o  o   r      n   n   T   r      e     e   e     e"
      write(*,*)"   T  o    o  r      n   n   T   r      eeeeee    eeeeee"
      write(*,*)"   T   o  o   r      n   n   T   r       e          e"
      write(*,*)"   T    oo    r      n   n   T   r        eeee       eeee"
      write(*,*)
      write(*,*)"*****************************************************"
      write(*,*)"*TornTree - A model to estimate tornado windspeeds   *"
      write(*,*)"*              from tree damage patterns.            *"
      write(*,*)"*                                                    *"
      write(*,*)"*Created by:  Andrew P. Holland - holland6@yahoo.com *"
      write(*,*)"*                                                    *"
      write(*,*)"* The TornTree model is used to estimate tornado     *"
      write(*,*)"*windspeeds from tree damage patterns.  It is the    *"
      write(*,*)"*combination of a vortex model (Peterson 1992) and a *"
      write(*,*)"*mechanistic tree model designed by Peltola et al.   *"
      write(*,*)"*The model calculates the windspeed and turning      * "
      write(*,*)"*moment at each tree.  If this turning moment is     *"
      write(*,*)"*greater than the resistance to stem breakage for    *"
      write(*,*)"*the tree then it is broken.  An output file is      *"
      write(*,*)"*created showing the orientation of the fallen trees.*"
      write(*,*)"*                                                    *"
      write(*,*)"*Peltola et al.,1999. Canadian J. of ForestResearch, *"
      write(*,*)"* 29, 647-661.                                       *"
      write(*,*)"*                                                    *"
      write(*,*)"* Peterson, R.E., 1992.  Weather and Forecasting, 7, *"
      write(*,*)"* 166-184.                                           *"
      write(*,*)"*                                                    *"
      write(*,*)"*****************************************************"

      write(*,2)'Type <return> to continue.'
 2    format(a,$)
      read(*,*)

      call mainmenu

      end
```

```
**MAINMENU************************************************************
*                                                                    *
*  Prompts user for the tree species they would like to run the model  *      *
on.                                                                *
*                                                                    *
**********************************************************************

      subroutine mainmenu
      character option        !the menu selection

      call clrscrn

      write(*,*)
      write(*,*)'***************************************************'
      write(*,*)'*                                                 *'
      write(*,*)'*(1)Run the model for Loblolly pine.              *'
      write(*,*)'*(2)Run the model for Scots pine.                 *'
      write(*,*)'*                                                 *'
      write(*,*)'***************************************************'
      write(*,3)
  3   format(/,'Enter an option (1-2) ->',$)
      read(*,4)option
  4   format(A)
      do while((option.lt.'1').or.(option.gt.'2'))
                write(*,3)
                read(*,4)option
      enddo

      if(option.eq.'1')then
          call loblollymenu(option)
      else if(option.eq.'2')then
        call scotspinemenu(option)
      end if

      return

      end


**LOBLOLLYMENU********************************************************
*                                                                    *
*  Prompts user for age class they would like to run the model on.   *
*                                                                    *
**********************************************************************

      subroutine loblollymenu(option)

      character selection     !the menu selection
      character option        !species option
      character answer        !choice between tornado or microburst
      real grid               !the tree spacing in meters
      real d                  !counter to determine how many trees
                                 !will be drawn
      real ran(300,300)       !a random number
      real vtorn(300,300)     !the vortex windspeed in m/s
      real ucom(300,300)      !u-component of vtorn (m/s)
      real vcom(300,300)      !v-component of vtorn (m/s)
```

```
      real gapsize             !size of the upwind gap in tree heights
      real psi(300,300)        !angle of vtorn from east
      write(*,*)
      write(*,*)'***************************************************'
      write(*,*)'*                                                 *'
      write(*,*)'* Site index is the height in feet of 50 year old  *'
      write(*,*)'* trees in this stand and is based on soil         *'
      write(*,*)'* properties.                                      *'
      write(*,*)'*                                                 *'
      write(*,*)'*(1)Site Index = 60                                *'
      write(*,*)'*(2)Site Index = 70                                *'
      write(*,*)'*(3)Site Index = 80                                *'
      write(*,*)'*(4)Site Index = 90                                *'
      write(*,*)'*(5)Site Index = 100                               *'
      write(*,*)'*(6)Site Index = 110                               *'
      write(*,*)'*(7)Site Index = 120                               *'
      write(*,*)'*(8)Input data for any loblolly pine tree.         *'
      write(*,*)'*                                                  *'
      write(*,*)'***************************************************'
      write(*,1)
    1 format(/,'Enter an option (1-8) ->',$)
      read(*,2)selection
    2 format(A)
      do while((selection.lt.'1').or.(selection.gt.'8'))
                write(*,1)
                read(*,2)selection
      enddo

*     Set grid spacing and counter values based on age class.

      if(selection.eq.'1')then
            grid=3.5
            d=2.
      write(*,*)"Enter a seed for the random number generator."
      write(*,*)"*This number can be any number of digits."
      write(*,*)"*This allows the trees to be randomly "
      write(*,*)"*distributed by age."
        read(*,*)ran(1,1)
      else if(selection.eq.'2')then
            grid=4.0
            d=2.
      write(*,*)"Enter a seed for the random number generator."
      write(*,*)"*This number can be any number of digits."
      write(*,*)"*This allows the trees to be randomly "
      write(*,*)"*distributed by age."
        read(*,*)ran(1,1)
      else if(selection.eq.'3')then
            grid=4.4
            d=2.0
      write(*,*)"Enter a seed for the random number generator."
      write(*,*)"*This number can be any number of digits."
      write(*,*)"*This allows the trees to be randomly "
      write(*,*)"*distributed by age."
        read(*,*)ran(1,1)
      else if(selection.eq.'4')then
            grid=5.0
            d=2.
```

```
        write(*,*)"Enter a seed for the random number generator."
        write(*,*)"*This number can be any number of digits."
        write(*,*)"*This allows the trees to be randomly "
        write(*,*)"*distributed by age."
          read(*,*)ran(1,1)
        else if(selection.eq.'5')then
              grid=5.1
              d=2.
        write(*,*)"Enter a seed for the random number generator."
        write(*,*)"*This number can be any number of digits."
        write(*,*)"*This allows the trees to be randomly "
        write(*,*)"*distributed by age."
          read(*,*)ran(1,1)
        else if(selection.eq.'6')then
              grid=5.4
              d=2.
        write(*,*)"Enter a seed for the random number generator."
        write(*,*)"*This number can be any number of digits."
        write(*,*)"*This allows the trees to be randomly "
        write(*,*)"*distributed by age."
          read(*,*)ran(1,1)
        else if(selection.eq.'7')then
              grid=5.7
              d=2.
        write(*,*)"Enter a seed for the random number generator."
        write(*,*)"*This number can be any number of digits."
        write(*,*)"*This allows the trees to be randomly "
        write(*,*)"*distributed by age."
          read(*,*)ran(1,1)
          else if(selection.eq.'8')then
              call loblollydata(grid)
              d=2.
        end if

*       Calculate the vortex windspeed.

        call vortex(grid,d,vtorn,ucom,vcom,gapsize,vmax,rc,vrmax,
       $      vforw,vtornmax,vtornmaxmph,answer,psi)

*       Calculate the critical wind speed with distance into the stand.
*       Since selections 1 through 7 are random plots, critical wind speed
*       cannot be calculated.

        if(selection.eq.'8')then
          call critwind(option,gapsize,grid,selection)
        end if

*       Calculate the turning moment and create output file.

        call loblolly(grid,d,ran,vtorn,ucom,vcom,gapsize,selection,
       $      vmax,rc,vrmax,vforw,vtornmax,vtornmaxmph,answer,psi)

        return

        end
```

```
**SCOTSPINEMENU*********************************************************
*                                                                       *
*  Prompts user for age class they would like to run the model on.     *
*                                                                       *
*************************************************************************
      subroutine scotspinemenu(option)

      character selection      !the menu selection
      character option         !species option
      character answer         !choice between tornado or microburst
      real grid                !the tree spacing in meters
      real d                   !counter to determine how many trees
                                  !will be drawn
      real ran(300,300)        !a random number
      real vtorn(300,300)      !the vortex windspeed in m/s
      real ucom(300,300)       !u-component of vtorn (m/s)
      real vcom(300,300)       !v-component of vtorn (m/s)
      real gapsize             !size of the upwind gap in tree heights
      real psi(300,300)        !angle of vtorn from east

      write(*,*)
      write(*,*)'*****************************************************'
      write(*,*)'*                                                   *'
      write(*,*)'*(1)All trees age 35   -> h=9.3m,  dbh=9.0cm        *'
      write(*,*)'*(2)All trees age 65   -> h=19.3m, dbh=17.4cm       *'
      write(*,*)'*(3)All trees age 95   -> h=25.0m, dbh=25.0cm       *'
      write(*,*)'*(4)All trees age 120  -> h=26.2m, dbh=28.5cm       *'
      write(*,*)'*(5)Trees randomly distributed from all age classes*'
      write(*,*)'*                                                   *'
      write(*,*)'*****************************************************'
      write(*,1)
  1   format(/,'Enter an option (1-5) ->',$)
      read(*,2)selection
  2   format(A)
      do while((selection.lt.'1').or.(selection.gt.'5'))
                write(*,1)
                read(*,2)selection
      enddo

*     Set grid spacing and counter values based on age class.

      if(selection.eq.'1')then
            grid=1.72
            d=4.0
      else if(selection.eq.'2')then
            grid=2.82
            d=3.0
      else if(selection.eq.'3')then
            grid=3.7
            d=2.0
      else if(selection.eq.'4')then
            grid=4.
            d=2.0
      else if(selection.eq.'5')then
            grid=3.7
            d=2.0
      write(*,*)"Enter a seed for the random number generator."
```

```
      write(*,*)"*This number can be any number of digits."
      write(*,*)"*This allows the trees to be randomly "
      write(*,*)"*distributed by age."
        read(*,*)ran(1,1)
      end if

*     Calculate the vortex windspeed.

      call vortex(grid,d,vtorn,ucom,vcom,gapsize,vmax,rc,vrmax,
     $      vforw,vtornmax,vtornmaxmph,answer,psi)

*     Calculate the critical wind speed with distance into the stand.
*     Since selection 5 is a random plot, critical wind speed cannot
*     be calculated.

      if(selection.eq.'1'.or.selection.eq.'2'.or.selection.eq.
     $      '3'.or.selection.eq.'4')then
        call critwind(option,gapsize,grid,selection)
      end if
      write(*,*)"Printing plot of fallen trees...(trees.ps)"
*     Calculate the turning moment and create output file.

      call scotspine(grid,d,ran,vtorn,ucom,vcom,gapsize,selection,
     $      vmax,rc,vrmax,vforw,vtornmax,vtornmaxmph,answer,psi)

      return

      end


**VORTEX*************************************************************
*                                                                   *
*  Calculates the vortex windspeed and generates a graphical output    *     *
file, 'tornado.ps.'                                              *
*                                                                   *
*********************************************************************

      subroutine vortex(grid,d,vtorn,ucom,vcom,gapsize,vmax,rc,
     $      vrmax,vforw,vtornmax,vtornmaxmph,answer,psi)

      real x(300,0:300),y(300,300)    !(x,y) coordinates of each point
      real r(300,300)                 !radius of each point in meters
      real vtan(300,300)              !tangential velocity at each point
                                          !(m/s)
      real theta(300,300),gamma(300,300) !angle of the tangential wind
                                           !measured from the x-axis
      real xss(300,300),yss(300,300)  !starting point of the arrow
      real xpp(300,300),ypp(300,300)  !end point of the arrow
      real ut(300,300),vt(300,300)    !vector components of tangential
                                         !wind (m/s)
      real uta(300,300),vta(300,300)  !ut and vt times 0.3
                                         !(scaling factor) for plots
      real vrad(300,300)              !radial velocity in m/s
      real delta(300,300)             !angle of the radial wind
                                         !measured from the x-axis
      real ur(300,300),vr(300,300)    !vector components of the radial
                                         !wind (m/s)
```

```
      real ura(300,300),vra(300,300)  !ur and vr times 0.3
      real utr(300,300),vtr(300,300)  !tangential plus radial wind
                                          !vectors
      real vtran(300,300),utran(300,300) !vector components of forward
                                           !motion (m/s)
      real ucom(300,300),vcom(300,300)   !forward motion plus utran and
                                            !vtran
      real vtorn(300,300)             !vector of vortex winds in m/s
      real rc                         !radius of the max winds in meters
      real vmax                       !maximum tangential velocity in
                                          !m/s
      real grid                       !tree spacing in meters
      real xg,yg                      !number of points in the grid
      real xmax,xmin,ymax,ymin        !max and min values of x and y
      real vforw                      !forward speed of the vortex in
                                           !m/s
      real vrmax                      !maximum radial wind speed in m/s
      real gapsize                    !size of the upwind gap in tree
                                           !heights
      real d                          !counter to determine number of
                                           !trees drawn
      real vtornmax                 !maximum vortex wind speed
      character choice              !choice of contour plot
      character answer              !choice of tornado or microburst
      real micromax                 !max. microburst wind speed
      real maxwind                  !max. microburst wind speed
      real psi(300,300)             !angle of vtorn from east

      xmax=200
      xmin=-200
      ymax=200
      ymin=-200
      pi=3.14

      write(*,*)'*****************************************************'
      write(*,*)'*                                                   *'
      write(*,*)'*    Choose one:                                    *'
      write(*,*)'*    (1)  Tornado                                   *'
      write(*,*)'*    (2)  Microburst                                *'
      write(*,*)'*                                                   *'
      write(*,*)'*****************************************************'
      write(*,1)
  1     format(/,'Enter an option (1-2) ->',$)
        read(*,2)answer
  2     format(A)
        do while((answer.lt.'1').or.(answer.gt.'2'))
                write(*,1)
                read(*,2)answer
        enddo

*       Read in input variables.

      if(answer.eq.'1')then
       write(*,*)
         write(*,*)"What is the maximum wind speed in m/s?"
       write(*,*)"*This is your initial guess.*"
         read(*,*)vmax
```

```
          write(*,*)"What is the radius of max winds in meters?"
          read(*,*)rc
          write(*,*)"What is the maximum  radial wind speed in m/s?"
          read(*,*)vrmax
          write(*,*)"What is the forward speed of the tornado in m/s?"
          read(*,*)vforw
       else
        write(*,*)
        write(*,*)"What is the maximum microburst velocity in m/s?"
        write(*,*)"*This is your initial guess.*"
        read(*,*)maxwind
        vrmax=-maxwind
        write(*,*)"What is the width of the damage path in meters?"
        read(*,*)rc
        write(*,*)"What is the forward speed of the thunderstorm"
        write(*,*)"in m/s?"
        read(*,*)vforw
        vmax=0.0
       end if

       write(*,*)"What is the size of the upwind gap in tree heights?"
       write(*,*)"* 0<gapsize<=10 *"
       read(*,*)gapsize
       do while((gapsize.le.0.).or.(gapsize.gt.10.))
         write(*,*)"What is the size of the upwind gap in tree heights?"
       write(*,*)"* 0<gapsize<=10  *"
       read(*,*)gapsize
         enddo
       write(*,*)"Would you like to print out a color contour plot"
       if(answer.eq.'1')then
       write(*,*)"of the vortex winds (y/n)?"
       else
       write(*,*)"of the microburst winds (y/n)?"
       end if
       read(*,*)choice

       xg=xmax*2/grid
       yg=ymax*2/grid

*        Calculate x.

         do 5 i=1,yg+1
         do 6 j=0
                 x(i,j)=xmin-grid
  6      continue
  5      continue

         do 7 i=1,yg+1
         do 8 j=1,xg+1
                 x(i,j)=x(i,j-1)+grid
  8      continue
  7      continue

*        Calculate y.

         do 9 i=1,yg+1
         do 10 j=1,xg+1
```

```
          if (i .eq. 1) then
                y(i,j)=ymax
          else if (i .ne. 1) then
                y(i,j)=y(i-1,j)-grid
          end if
 10     continue
 9      continue

*       Calculate radius to each grid point.

        do 30 i=1,yg+1
        do 30 j=1,xg+1
                r(i,j)=sqrt((x(i,j)**2)+(y(i,j)**2))
 30     continue

*       Calculate tangential velocities.

        do 50 i=1,yg+1
        do 50 j=1,xg+1
        if (r(i,j) .ge. rc) then
                vtan(i,j)=(vmax*rc)/r(i,j)
        else if (r(i,j) .lt. rc) then
                vtan(i,j)=(vmax*r(i,j))/rc
        end if
 50     continue

*       Calculate theta.

        do 70 i=1,yg+1
        do 70 j=1,xg+1
        if (x(i,j) .ne. 0 .and. y(i,j) .ne. 0) then
                theta(i,j)=(atan(y(i,j)/x(i,j))*(180/pi))
        else if (x(i,j) .eq. 0 .and. y(i,j) .eq. 0) then
                theta(i,j)=0.0
        end if
 70     continue

*       Calculate gamma (angle of tangential winds).

        do 85 i=1,yg+1
        do 85 j=1,xg+1
        if (x(i,j) .gt. 0 .and. y(i,j) .gt. 0) then
                gamma(i,j)=90+theta(i,j)
        else if (x(i,j) .lt. 0 .and. y(i, j) .gt .0) then
                gamma(i,j)=270+theta(i,j)
        else if (x(i,j) .lt. 0 .and. y(i,j) .lt. 0) then
                gamma(i,j)=theta(i,j)-90
        else if (x(i,j) .gt. 0 .and. y(i,j) .lt .0) then
                gamma(i,j)=90+theta(i,j)
        else if (x(i,j) .eq. 0 .and. y(i,j) .gt. 0) then
                gamma(i,j)=180
        else if (x(i,j) .eq. 0 .and. y(i,j) .lt. 0) then
                gamma(i,j)=0
        else if (x(i,j) .gt. 0 .and. y(i,j) .eq. 0) then
                gamma(i,j)=90
        else if (x(i,j) .lt. 0 .and. y(i,j) .eq. 0) then
                gamma(i,j)=270
```

```
      end if
 85   continue

*     Move origin to middle of the page.

      do 95 i=1,yg+1
      do 95 j=1,xg+1
            xss(i,j)=x(i,j)+200
            yss(i,j)=y(i,j)+200
 95   continue

*     Calculate the vector components of the tangential wind.

      do 100 i=1,yg+1
      do 100 j=1,xg+1
            ut(i,j)=vtan(i,j)*(cos(gamma(i,j)*pi/180))
            vt(i,j)=vtan(i,j)*(sin(gamma(i,j)*pi/180))
            uta(i,j)=ut(i,j)*.3
            vta(i,j)=vt(i,j)*.3
100   continue

*     Calculate radial velocities.

      do 120 i=1,yg+1
      do 120 j=1,xg+1
      if (r(i,j) .ge. rc) then
            vrad(i,j)=(vrmax*rc)/r(i,j)
      else if (r(i,j) .lt. rc) then
            vrad(i,j)=(vrmax*r(i,j))/rc
      end if
120   continue

*     Calculate delta (angle of inflow).

      do 121 i=1,yg+1
      do 121 j=1,xg+1
      if (x(i,j) .gt. 0 .and. y(i,j) .gt. 0) then
            delta(i,j)=180+theta(i,j)
      else if (x(i,j) .lt. 0 .and. y(i,j) .gt .0) then
            delta(i,j)=theta(i,j)
      else if (x(i,j) .lt. 0 .and. y(i,j) .lt. 0) then
            delta(i,j)=theta(i,j)
      else if (x(i,j) .gt. 0 .and. y(i,j) .lt .0) then
            delta(i,j)=180+theta(i,j)
      else if (x(i,j) .eq. 0 .and. y(i,j) .gt. 0) then
            delta(i,j)=270
      else if (x(i,j) .eq. 0 .and. y(i,j)  .lt. 0) then
            delta(i,j)=90
      else if (x(i,j) .gt. 0 .and. y(i,j) .eq. 0) then
            delta(i,j)=180
      else if (x(i,j) .lt. 0 .and. y(i,j) .eq. 0) then
            delta(i,j)=0
      end if
121   continue

*     Calculate the vector components of the radial wind.
```

```
        do 130 i=1,yg+1
        do 130 j=1,xg+1
                ur(i,j)=vrad(i,j)*(cos(delta(i,j)*pi/180))
                vr(i,j)=vrad(i,j)*(sin(delta(i,j)*pi/180))
                ura(i,j)=ur(i,j)*.3
                vra(i,j)=vr(i,j)*.3
                utr(i,j)=uta(i,j)+ura(i,j)
                vtr(i,j)=vta(i,j)+vra(i,j)
  130   continue


*       Print graph of moving tornado.
      write(*,*)"Printing vortex wind field...(tornado.ps)"
      call newdev('tornado.ps',10)
      call psinit(.true.)
      call factor(0.6)
      call plot(4.,120.,-3)
      call border(300.,300.,0000,1111,4,2,3,4)
      call keknum(-1.,150.,5.,0.,90.,-1,1)
      call keknum(-1.,300.,5.,200.,90.,-1,1)
      call keknum(-1.,-8.,5.,-200.,0.,-1,2)
      call keknum(150.,-8.,5.,0.,0.,-1,1)
      call keknum(300.,-8.,5.,200.,0.,-1,1)
      call keksym(150.,-20.,7.,12hDistance (m),0.,12,1)
      call keksym(-8.,150.,7.,12hDistance (m),90.,12,1)
      call setlw(1.0)
      call arrow(150.,305.,150.,330.,5.,30.,0)
      call setlw(0.)
      if(answer.eq.'1')then
      call keksym(160.,320.,6.,7hTornado,0.,7,0)
      call keksym(160.,310.,6.,8hMovement,0.,8,0)
      else
      call keksym(160.,320.,6.,12hThunderstorm,0.,12,0)
      call keksym(160.,310.,6.,8hMovement,0.,8,0)
      end if
      call factor(0.3)
      call setlw(2.0)
      call arrow(615.,190.,640.,190.,5.,30.,0)
      call setlw(0.)
      call keksym(627.,200.,10.,6hScale:,0.,6,1)
      call keknum(627.,170.,10.,60.,0.,-1,1)
      call keksym(627.,157.,10.,3hm/s,0.,3,1)
      call plot(0.,0.,-3)
      call factor(0.45)
        do 135 i=yg+1,1,-d
        do 135 j=1,xg+1,d
                utran(i,j)=0
                vtran(i,j)=vforw*(sin(5156.6202*pi/180))
                vtran(i,j)=vtran(i,j)*.3
                ucom(i,j)=utran(i,j)+utr(i,j)
                vcom(i,j)=vtran(i,j)+vtr(i,j)
                vtorn(i,j)=sqrt(((ucom(i,j)/.3)**2)+((vcom(i,j)/.3)**2))
                xpp(i,j)=xss(i,j)+ucom(i,j)
                ypp(i,j)=yss(i,j)+vcom(i,j)
                call arrow(xss(i,j),yss(i,j),xpp(i,j),ypp(i,j),3.,30.,0)
                psi(i,j)=atand(vcom(i,j)/ucom(i,j))
                if(ucom(i,j).ge.0.and.vcom(i,j).ge.0)then
```

```
                psi(i,j)=psi(i,j)
              else if(ucom(i,j).ge.0.and.vcom(i,j).lt.0)then
               psi(i,j)=360+psi(i,j)
            else if(ucom(i,j).lt.0.and.vcom(i,j).ge.0)then
               psi(i,j)=180+psi(i,j)
            else if(ucom(i,j).lt.0.and.vcom(i,j).lt.0)then
               psi(i,j)=180+psi(i,j)
            end if
  135   continue
      call factor(0.45)

*     Obtain max value of vortex winds and convert from m/s to mph.

      vtornmax=-1.e20
      do j=1,xg+1,d
       do i=1,yg+1,d
            vtornmax=amax1(vtornmax,vtorn(i,j))
       enddo
      enddo
      vtornmax=int(vtornmax)
      vtornmaxmph=vtornmax*2.237

      if(answer.eq.'1')then
      call keksym(10.,-40.,7.,27hMax. tangential velocity = ,0.,
     $      27,0)
      call keknum(999.,999.,7.,vmax,0.,-1,0)
      call keksym(999.,999.,7.,4h m/s,0.,4,0)
      call keksym(10.,-50.,7.,23hRadius of max. winds = ,0.,23,0)
      call keknum(999.,999.,7.,rc,0.,-1,0)
      call keksym(999.,999.,7.,2h m,0.,2,0)
      call keksym(10.,-60.,7.,23hMax. radial velocity = ,0.,23,0)
      call keknum(999.,999.,7.,vrmax,0.,-1,0)
      call keksym(999.,999.,7.,4h m/s,0.,4,0)
      call keksym(10.,-70.,7.,16hForward speed = ,0.,16,0)
      call keknum(999.,999.,7.,vforw,0.,-1,0)
      call keksym(999.,999.,7.,4h m/s,0.,4,0)
      call keksym(10.,-80.,7.,21hMaximum wind speed = ,0.,21,0)
      call keknum(999.,999.,7.,vtornmax,0.,-1,0)
      call keksym(999.,999.,7.,7h m/s = ,0.,7,0)
      call keknum(999.,999.,7.,vtornmaxmph,0.,-1,0)
      call keksym(999.,999.,7.,4h mph,0.,4,0)
      else
      call keksym(10.,-40.,7.,27hMax. microburst velocity = ,0.,
     $      27,0)
      micromax=-vrmax
      call keknum(999.,999.,7.,micromax,0.,-1,0)
      call keksym(999.,999.,7.,4h m/s,0.,4,0)
      call keksym(10.,-50.,7.,23hWidth of damage path = ,0.,23,0)
      call keknum(999.,999.,7.,rc,0.,-1,0)
      call keksym(999.,999.,7.,2h m,0.,2,0)
      call keksym(10.,-60.,7.,16hForward speed = ,0.,16,0)
      call keknum(999.,999.,7.,vforw,0.,-1,0)
      call keksym(999.,999.,7.,4h m/s,0.,4,0)
      call keksym(10.,-70.,7.,21hMaximum wind speed = ,0.,21,0)
      call keknum(999.,999.,7.,vtornmax,0.,-1,0)
      call keksym(999.,999.,7.,7h m/s = ,0.,7,0)
      call keknum(999.,999.,7.,vtornmaxmph,0.,-1,0)
```

```
      call keksym(999.,999.,7.,4h mph,0.,4,0)
      end if
      if(answer.eq.'1')then
      if((vtornmaxmph.ge.40.).and.(vtornmaxmph.le.72.))then
      call keksym(10.,-90.,7.,10hF0 Tornado,0.,10,0)
      else if((vtornmaxmph.ge.73.).and.(vtornmaxmph.le.112.))then
      call keksym(10.,-90.,7.,10hF1 Tornado,0.,10,0)
      else if((vtornmaxmph.ge.113.).and.(vtornmaxmph.le.157.))then
      call keksym(10.,-90.,7.,10hF2 Tornado,0.,10,0)
      else if((vtornmaxmph.ge.158.).and.(vtornmaxmph.le.206.))then
      call keksym(10.,-90.,7.,10hF3 Tornado,0.,10,0)
      else if((vtornmaxmph.ge.207.).and.(vtornmaxmph.le.260.))then
      call keksym(10.,-90.,7.,10hF4 Tornado,0.,10,0)
      else if((vtornmaxmph.ge.261.).and.(vtornmaxmph.le.318.))then
      call keksym(10.,-90.,7.,10hF5 Tornado,0.,10,0)
      else if(vtornmaxmph.ge.319)then
      call keksym(10.,-90.,7.,10hF6 Tornado,0.,10,0)
      end if
      else
      call keksym(10.,-90.,7.,10hMicroburst,0.,10,0)
      end if
      call keksym(275.,-125.,5.,8hTornTree,0.,8,0)
      call plotnd

*     Make color contour plot if user chooses to do so.

      if(choice.eq.'y')then
            call colorplot(d,grid,ucom,vcom,vmax,rc,
     $            vrmax,vforw,vtornmax,vtornmaxmph,xss,yss,answer)

      end if

      return
      end


**LOBLOLLY*************************************************************
*                                                                     *
*  Calculates the turning moment and generates a graphical output file *
*  'trees.ps.'                                                        *
*                                                                     *
***********************************************************************

      subroutine loblolly(grid,d,ran,vtorn,ucom,vcom,gapsize,
     $      selection,vmax,rc,vrmax,vforw,vtornmax,vtornmaxmph,answer,psi)

      real x(300,0:300),y(300,300)    !(x,y) coordinates of each point
      real xss(300,300),yss(300,300)  !starting point of the arrow
      real xpp(300,300),ypp(300,300)  !end point of the arrow
      real z(0:50)                    !height of each one-meter segment
                                           !(m)
      real vtree(300,300,0:50)        !3-D wind field of vortex (m/s)
      real ran(300,300)               !random number
      real age(300,300)               !randomly determined age class of
                                           !tree
      real rownum(300,300)            !number of row in the forest
      real distedge(300,300)          !distance from forest edge in tree
```

```
                                     !heights
real gustmax(300,300)           !used to calculate gust factor
real gustmeanedge(300,300)      !used to calculate gust factor
real gustfac(300,300)           !factor to include gustiness of
                                     !the wind
real gapmax(300,300)            !used to calculate gap factor
real gapmeanedge(300,300)       !used to calculate gap factor
real gapfac(300,300)            !factor to include size of the
                                     !upwind gap
real stemres(300,300)           !resistance to stem breakage (Nm)
real area(300,300,0:50)         !area of each one-meter segment
                                     !(m^2)
real ms(300,300,0:50)           !mass of each one-meter segment
                                     (kg)
real fwind(300,300,0:50)        !force due to the wind (N)
real fwindtot(300,300)          !sum of fwind over the tree height
                                     !(N)
real fmass(300,300,0:50)        !force due to gravity (N)
real ltop(0:50)                 !distance from tree top (m)
real horizd(300,300,0:50)       !horizontal displacement of the
                                     !stem (m)

real h(300,300)                 !height of the tree (m)
real cd                         !drag coefficient
real rho                        !density of air (kg/m^3)
real g                          !gravitational constant (m/s^2)
real dbh(300,300)               !diameter at breast height (m)
real a(300,300)                 !distance from ground to crown
                                     !center (m)
real b(300,300)                 !distance from tree top to crown
                                     !center (m)
real crhgt(300,300)             !height from ground to crown base
                                     !(m)
real stemhgt(300,300)           !total height - crhgt, height of
                                     !stem (m)
real moe                        !modulus of elasticity (Pa)
real mor                        !modulus of rupture (Pa)
real ami(300,300)               !area moment of inertia (m^4)
real gh(300,300)                !spacing between trees in tree
                                     !heights
real grid                       !tree spacing (m)
real xg,yg                      !number of points in the grid
real xmax,xmin,ymax,ymin        !max and min values of x and y
real gapsize                    !size of the upwind gap in tree
                                     !heights
real d                          !counter to determine number of
                                     !trees drawn
real ucrarea(300,300,0:50)      !unstreamlined crown area (m^2 )
real starea(300,300,0:50)       !stem area (m^2)
real crarea(300,300,0:50)       !streamlined crown area (m^2)
real st(300,300,0:50)           !streamlining function
real ucom(300,300),vcom(300,300) !forward motion plus utran and
                                     !vtran (m/s)
real vtorn(300,300)             !vector of vortex winds (m/s)
real vtornmax                   !max vortex wind speed (m/s)
real ht,db                      !numbers used to get the height
```

```
                                                    !and dbh maximums

      character selection              !the menu selection
      character answer               !choice between tornado or
                                              !microburst
      real tmax(300,300)             !maximum turning moment (Nm)
      real tmom(300,300,0:50)        !turning moment for segment (Nm)
      real micromax                  !max. microburst wind speed (m/s)
      real psi(300,300)              !angle of vtorn from the east
      real uc(300,300)               !used to make the length of the
                                              !fallen tree vector
                                              !proportional to the tree
                                              !height
      real vc(300,300)               !"
      integer damage(300,300)        !0 for standing tree, 1 for fallen tree
      real stand                     !number of trees in the stand
      real fallen                    !number of fallen trees
      real percent                   !percent of fallen trees

      xmax=200
      xmin=-200
      ymax=200
      ymin=-200
      pi=3.14
      cd=0.2
      rho=1.226
      g=9.8
      moe=2.41E9
      mor=49000000.

      xg=xmax*2/grid
      yg=ymax*2/grid

*       Calculate x and y for the tree grid.

        do 400 m=1,yg+1
        do 400 n=0
                x(m,n)=xmin-grid
  400   continue

        do 410 m=1,yg+1
        do 410 n=1,xg+1
                x(m,n)=x(m,n-1)+grid
  410   continue

        do 420 m=1,yg+1
        do 420 n=1,xg+1
                if (m .eq. 1) then
                        y(m,n)=ymax
                else if (m .ne. 1) then
                        y(m,n)=y(m-1,n)-grid
                end if
  420   continue

        do 425 m=1,yg+1
        do 425 n=1,xg+1
                xss(m,n)=x(m,n)+200
```

```
                yss(m,n)=y(m,n)+200
  425   continue

*       Randomly pick age class of each tree.

        do 430 m=1,yg+1
        do 430 n=1,xg+1
        call random(ran(m,n))
        if (ran(m,n) .ge. 0.000 .and. ran(m,n) .le. 0.143)then
                                age(m,n)=1.
        else if(ran(m,n).gt.0.143000001 .and. ran(m,n).le.0.286)then
                                age(m,n)=2.
        else if(ran(m,n).gt.0.286000001 .and. ran(m,n).le.0.429)then
                                age(m,n)=3.
        else if(ran(m,n).gt.0.429000001 .and. ran(m,n).le.0.572)then
                                age(m,n)=4.
        else if(ran(m,n).gt.0.572000001 .and. ran(m,n).le.0.715)then
                                age(m,n)=5.
        else if(ran(m,n).gt.0.715000001 .and. ran(m,n).le.0.858)then
                                age(m,n)=6.
        else if(ran(m,n).gt.0.858000001 .and. ran(m,n).le.1.000)then
                                age(m,n)=7.
        end if
  430   continue

*       Assign each row of the forest a number.

        do 456 m=1,yg+1
        do 456 n=1,xg+1
                rownum(m,n)=yg+1-m
  456   continue

*       Read in values of dbh and tree height based on menu choice.
*       'lp**age50.dat' -> Height and dbh values for ** Site Index.

        do 457 m=1,yg+1
        do 457 n=1,xg+1
          if(selection.eq.'1')then
          if(age(m,n).eq.1)then
                open(1,file='lp60age20.dat')
                read(1,*)dbh(m,n),h(m,n)
                close(1)
          else if(age(m,n).eq.2)then
                open(2,file='lp60age30.dat')
                read(2,*)dbh(m,n),h(m,n)
                close(2)
          else if(age(m,n).eq.3)then
                open(3,file='lp60age40.dat')
                read(3,*)dbh(m,n),h(m,n)
                close(3)
          else if(age(m,n).eq.4)then
                open(4,file='lp60age50.dat')
                read(4,*)dbh(m,n),h(m,n)
                close(4)
          else if(age(m,n).eq.5)then
                open(5,file='lp60age60.dat')
                read(5,*)dbh(m,n),h(m,n)
```

```
      close(5)
else if(age(m,n).eq.6)then
      open(6,file='lp60age70.dat')
      read(6,*)dbh(m,n),h(m,n)
      close(6)
else if(age(m,n).eq.7)then
      open(7,file='lp60age80.dat')
      read(7,*)dbh(m,n),h(m,n)
      close(7)
end if
else if(selection.eq.'2')then
  if(age(m,n).eq.1)then
      open(1,file='lp70age20.dat')
      read(1,*)dbh(m,n),h(m,n)
      close(1)
else if(age(m,n).eq.2)then
      open(2,file='lp70age30.dat')
      read(2,*)dbh(m,n),h(m,n)
      close(2)
else if(age(m,n).eq.3)then
      open(3,file='lp70age40.dat')
      read(3,*)dbh(m,n),h(m,n)
      close(3)
else if(age(m,n).eq.4)then
      open(4,file='lp70age50.dat')
      read(4,*)dbh(m,n),h(m,n)
      close(4)
else if(age(m,n).eq.5)then
      open(5,file='lp70age60.dat')
      read(5,*)dbh(m,n),h(m,n)
      close(5)
else if(age(m,n).eq.6)then
      open(6,file='lp70age70.dat')
      read(6,*)dbh(m,n),h(m,n)
      close(6)
else if(age(m,n).eq.7)then
      open(7,file='lp70age80.dat')
      read(7,*)dbh(m,n),h(m,n)
      close(7)
end if
else if(selection.eq.'3')then
  if(age(m,n).eq.1)then
      open(1,file='lp80age20.dat')
      read(1,*)dbh(m,n),h(m,n)
      close(1)
else if(age(m,n).eq.2)then
      open(2,file='lp80age30.dat')
      read(2,*)dbh(m,n),h(m,n)
      close(2)
else if(age(m,n).eq.3)then
      open(3,file='lp80age40.dat')
      read(3,*)dbh(m,n),h(m,n)
      close(3)
else if(age(m,n).eq.4)then
      open(4,file='lp80age50.dat')
      read(4,*)dbh(m,n),h(m,n)
      close(4)
```

```
  else if(age(m,n).eq.5)then
        open(5,file='lp80age60.dat')
        read(5,*)dbh(m,n),h(m,n)
        close(5)
  else if(age(m,n).eq.6)then
        open(6,file='lp80age70.dat')
        read(6,*)dbh(m,n),h(m,n)
        close(6)
  else if(age(m,n).eq.7)then
        open(7,file='lp80age80.dat')
        read(7,*)dbh(m,n),h(m,n)
        close(7)
  end if
else if(selection.eq.'4')then
    if(age(m,n).eq.1)then
        open(1,file='lp90age20.dat')
        read(1,*)dbh(m,n),h(m,n)
        close(1)
  else if(age(m,n).eq.2)then
        open(2,file='lp90age30.dat')
        read(2,*)dbh(m,n),h(m,n)
        close(2)
  else if(age(m,n).eq.3)then
        open(3,file='lp90age40.dat')
        read(3,*)dbh(m,n),h(m,n)
        close(3)
  else if(age(m,n).eq.4)then
        open(4,file='lp90age50.dat')
        read(4,*)dbh(m,n),h(m,n)
        close(4)
  else if(age(m,n).eq.5)then
        open(5,file='lp90age60.dat')
        read(5,*)dbh(m,n),h(m,n)
        close(5)
  else if(age(m,n).eq.6)then
        open(6,file='lp90age70.dat')
        read(6,*)dbh(m,n),h(m,n)
        close(6)
  else if(age(m,n).eq.7)then
        open(7,file='lp90age80.dat')
        read(7,*)dbh(m,n),h(m,n)
        close(7)
  end if
else if(selection.eq.'5')then
    if(age(m,n).eq.1)then
        open(1,file='lp100age20.dat')
        read(1,*)dbh(m,n),h(m,n)
        close(1)
  else if(age(m,n).eq.2)then
        open(2,file='lp100age30.dat')
        read(2,*)dbh(m,n),h(m,n)
        close(2)
  else if(age(m,n).eq.3)then
        open(3,file='lp100age40.dat')
        read(3,*)dbh(m,n),h(m,n)
        close(3)
  else if(age(m,n).eq.4)then
```

```
        open(4,file='lp100age50.dat')
        read(4,*)dbh(m,n),h(m,n)
        close(4)
  else if(age(m,n).eq.5)then
        open(5,file='lp100age60.dat')
        read(5,*)dbh(m,n),h(m,n)
        close(5)
  else if(age(m,n).eq.6)then
        open(6,file='lp100age70.dat')
        read(6,*)dbh(m,n),h(m,n)
        close(6)
  else if(age(m,n).eq.7)then
        open(7,file='lp100age80.dat')
        read(7,*)dbh(m,n),h(m,n)
        close(7)
  end if
else if(selection.eq.'6')then
    if(age(m,n).eq.1)then
        open(1,file='lp110age20.dat')
        read(1,*)dbh(m,n),h(m,n)
        close(1)
  else if(age(m,n).eq.2)then
        open(2,file='lp110age30.dat')
        read(2,*)dbh(m,n),h(m,n)
        close(2)
  else if(age(m,n).eq.3)then
        open(3,file='lp110age40.dat')
        read(3,*)dbh(m,n),h(m,n)
        close(3)
  else if(age(m,n).eq.4)then
        open(4,file='lp110age50.dat')
        read(4,*)dbh(m,n),h(m,n)
        close(4)
  else if(age(m,n).eq.5)then
        open(5,file='lp110age60.dat')
        read(5,*)dbh(m,n),h(m,n)
        close(5)
  else if(age(m,n).eq.6)then
        open(6,file='lp110age70.dat')
        read(6,*)dbh(m,n),h(m,n)
        close(6)
  else if(age(m,n).eq.7)then
        open(7,file='lp110age80.dat')
        read(7,*)dbh(m,n),h(m,n)
        close(7)
  end if
else if(selection.eq.'7')then
    if(age(m,n).eq.1)then
        open(1,file='lp120age20.dat')
        read(1,*)dbh(m,n),h(m,n)
        close(1)
  else if(age(m,n).eq.2)then
        open(2,file='lp120age30.dat')
        read(2,*)dbh(m,n),h(m,n)
        close(2)
  else if(age(m,n).eq.3)then
        open(3,file='lp120age40.dat')
```

```
                read(3,*)dbh(m,n),h(m,n)
                close(3)
          else if(age(m,n).eq.4)then
                open(4,file='lp120age50.dat')
                read(4,*)dbh(m,n),h(m,n)
                close(4)
          else if(age(m,n).eq.5)then
                open(5,file='lp120age60.dat')
                read(5,*)dbh(m,n),h(m,n)
                close(5)
          else if(age(m,n).eq.6)then
                open(6,file='lp120age70.dat')
                read(6,*)dbh(m,n),h(m,n)
                close(6)
          else if(age(m,n).eq.7)then
                open(7,file='lp120age80.dat')
                read(7,*)dbh(m,n),h(m,n)
                close(7)
          end if
          else if(selection.eq.'8')then
                open(1,file='lpage.dat')
                read(1,*)dbh(m,n),h(m,n)
                close(1)
        end if
        gh(m,n)=grid/h(m,n)
        ami(m,n)=pi*((dbh(m,n)**4)/64)
        crhgt(m,n)=(h(m,n)*0.40)
        stemhgt(m,n)=h(m,n)-crhgt(m,n)
        a(m,n)=(crhgt(m,n)/2)+stemhgt(m,n)
        b(m,n)=h(m,n)-a(m,n)
  457   continue

*     Calulate the gust and gap factors and the tree's resistance to
*     stem breakage.

        do 470 m=yg+1,1,-1
        do 470 n=1,xg+1
        distedge(m,n)=gh(m,n)*rownum(m,n)
        gustmeanedge(m,n)=(((1.7239*gh(m,n)+0.0316)**(0.))
     $  *(-0.68*gh(m,n)+0.4785))+(0.68*gh(m,n)-0.0385)
        gustmax(m,n)=(((1.1127*gh(m,n)+0.0311)**(distedge(m,n)))
     $  *(-1.273*gh(m,n)+0.9701))+(2.7193*gh(m,n)-0.061)
        gustfac(m,n)=gustmax(m,n)/gustmeanedge(m,n)
        gapmeanedge(m,n)=(0.001+0.001*(10.**0.562))/0.00465
        gapmax(m,n)=(0.0072+0.0064*(gapsize**0.3467))/0.0214
        gapfac(m,n)=gapmax(m,n)/gapmeanedge(m,n)
        stemres(m,n)=(pi/32)*mor*(dbh(m,n)**3)
  470   continue

*     Read in the area and mass distributions of each tree.
*     'lp90ucrarea**.dat' -> Unstreamlined crown area distribution.
*     'lp90starea.dat' -> Stem area distribution.
*     'lp90mass.dat' -> Stem and crown mass distribution.

      do 475 m=1,yg+1
      do 475 n=1,xg+1
      do 477 k=0,h(m,n)-1
```

```
if(selection.eq.'1')then
  if(age(m,n).eq.1)then
        open(8,file='lp60ucrarea20.dat')
        read(8,*)ucrarea(m,n,k)
        open(9,file='lp60starea20.dat')
        read(9,*)starea(m,n,k)
        open(10,file='lp60mass20.dat')
        read(10,*)ms(m,n,k)
  else if(age(m,n).eq.2)then
        open(11,file='lp60ucrarea30.dat')
        read(11,*)ucrarea(m,n,k)
        open(12,file='lp60starea30.dat')
        read(12,*)starea(m,n,k)
        open(13,file='lp60mass30.dat')
        read(13,*)ms(m,n,k)
  else if(age(m,n).eq.3)then
        open(14,file='lp60ucrarea40.dat')
        read(14,*)ucrarea(m,n,k)
        open(15,file='lp60starea40.dat')
        read(15,*)starea(m,n,k)
        open(16,file='lp60 mass40.dat')
        read(16,*)ms(m,n,k)
  else if(age(m,n).eq.4)then
        open(17,file='lp60ucrarea50.dat')
        read(17,*)ucrarea(m,n,k)
        open(18,file='lp60starea50.dat')
        read(18,*)starea(m,n,k)
        open(19,file='lp60mass50.dat')
        read(19,*)ms(m,n,k)
  else if(age(m,n).eq.5)then
        open(20,file='lp60ucrarea60.dat')
        read(20,*)ucrarea(m,n,k)
        open(21,file='lp60starea60.dat')
        read(21,*)starea(m,n,k)
        open(22,file='lp60mass60.dat')
        read(22,*)ms(m,n,k)
  else if(age(m,n).eq.6)then
        open(23,file='lp60ucrarea70.dat')
        read(23,*)ucrarea(m,n,k)
        open(24,file='lp60star ea70.dat')
        read(24,*)starea(m,n,k)
        open(25,file='lp60mass70.dat')
        read(25,*)ms(m,n,k)
  else if(age(m,n).eq.7)then
        open(26,file='lp60ucrarea80.dat')
        read(26,*)ucrarea(m,n,k)
        open(27,file='lp60starea80.dat')
        read(27,*)starea(m,n,k)
        open(28,file='lp60mass80.dat')
        read(28,*)ms(m,n,k)
  end if
  else if(selection.eq.'2')then
    if(age(m,n).eq.1)then
        open(8,file='lp70ucrarea20.dat')
        read(8,*)ucrarea(m,n,k)
        open(9,file='lp70starea20.dat')
        read(9,*)starea(m,n,k)
```

```
      open(10,file='lp70mass20.dat')
      read(10,*)ms(m,n,k)
else if(age(m,n).eq.2 )then
      open(11,file='lp70ucrarea30.dat')
      read(11,*)ucrarea(m,n,k)
      open(12,file='lp70starea30.dat')
      read(12,*)starea(m,n,k)
      open(13,file='lp70mass30.dat')
      read(13,*)ms(m,n,k)
else if(age(m,n).eq.3)then
      open(14,file='lp70ucrarea40.dat')
      read(14,*)ucrarea(m,n,k)
      open(15,file='lp70starea40.dat')
      read(15,*)starea(m,n,k)
      open(16,file='lp70mass40.dat')
      read(16,*)ms(m,n,k)
else if(age(m,n).eq.4)then
      open(17,file='lp70ucrarea50.dat')
      read(17,*)ucrarea(m,n,k)
      open(18,file='lp70starea50.dat')
      read(18,*)starea(m,n,k)
      open(19,file='lp70mass50 .dat')
      read(19,*)ms(m,n,k)
else if(age(m,n).eq.5)then
      open(20,file='lp70ucrarea60.dat')
      read(20,*)ucrarea(m,n,k)
      open(21,file='lp70starea60.dat')
      read(21,*)starea(m,n,k)
      open(22,file='lp70mass60.dat')
      read(22,*)ms(m,n,k)
else if(age(m,n).eq.6)then
      open(23,file='lp70ucrarea70.dat')
      read(23,*)ucrarea(m,n,k)
      open(24,file='lp70starea70.dat')
      read(24,*)starea(m,n,k)
      open(25,file='lp70mass70.dat')
      read(25,*)ms(m,n,k)
else if(age(m,n).eq.7)then
      open(26,file='lp70ucrarea80.dat')
      read(26,*)ucrarea(m,n,k)
      open(27,file='lp70starea80.d at')
      read(27,*)starea(m,n,k)
      open(28,file='lp70mass80.dat')
      read(28,*)ms(m,n,k)
end if
else if(selection.eq.'3')then
  if(age(m,n).eq.1)then
      open(8,file='lp80ucrarea20 .dat')
      read(8,*)ucrarea(m,n,k)
      open(9,file='lp80starea20.dat')
      read(9,*)starea(m,n,k)
      open(10,file='lp80mass20.dat')
      read(10,*)ms(m,n,k)
else if(age(m,n).eq.2)then
      open( 11,file='lp80ucrarea30.dat')
      read(11,*)ucrarea(m,n,k)
      open(12,file='lp80starea30.dat')
```

```
            read(12,*)starea(m,n,k)
            open(13,file='lp80mass30.dat')
            read(13,*)ms(m,n,k)
      else if(age(m,n).eq.3)then
            open(14,file='lp80ucrarea40.dat')
            read(14,*)ucrarea(m,n,k)
            open(15,file='lp80starea40.dat')
            read(15,*)starea(m,n,k)
            open(16,file='lp80mass40.dat')
            read(16,*)ms(m,n,k)
      else if(age(m,n).eq.4)then
            open(17,file='lp80ucrarea50.dat')
            read(17,*)ucrarea(m,n,k)
            open(18,file='lp80starea50.dat')
            read(18,*)starea(m,n,k)
            open(19,file='lp80mass50.dat')
            read(19,*)ms(m,n,k)
      else if(age(m,n).eq.5)then
            open(20,file='lp80ucrarea60.dat')
            read(20,*)ucrarea(m,n,k)
            open(21,file='lp80starea60.dat')
            read(21,*)starea(m,n,k)
            open(22,file='lp80mass60.dat')
            read(22,*)ms(m,n,k)
      else if(age(m,n).eq.6)then
            open(23,file='lp80ucrarea70.dat')
            read(23,*)ucrarea(m,n,k)
            open(24,file='lp80starea70.dat')
            read(24,*)starea(m,n,k)
            open(25,file='lp80mass70.dat')
            read(25,*)ms(m,n,k)
      else if(age(m,n).eq.7)then
            open(26,file='lp80ucrarea80.dat')
            read(26,*)ucrarea(m,n,k)
            open(27,file='lp80starea80.dat')
            read(27,*)starea(m,n,k)
            open(28,file='lp80mass80.dat')
            read(28,*)ms(m,n,k)
      end if
    else if(selection.eq.'4')then
        if(age(m,n).eq.1)then
            open(8,file='lp90ucrarea20.dat')
            read(8,*)ucrarea(m,n,k)
            open(9,file='lp90starea20.dat')
            read(9,*)starea(m,n,k)
            open(10,file='lp90mass20.dat')
            read(10,*)ms(m,n,k)
      else if(age(m,n).eq.2)then
            open(11,file='lp90ucrarea30.dat')
            read(11,*)ucrarea(m,n,k)
            open(12,file='lp90starea30.dat')
            read(12,*)starea(m,n,k)
            open(13,file='lp90mass30.dat')
            read(13,*)ms(m,n,k)
      else if(age(m,n).eq.3)then
            open(14,file='lp90ucrarea40.dat')
            read(14,*)ucrarea(m,n,k)
```

```
        open(15,file='lp90starea40.dat')
        read(15,*)starea(m,n,k)
        open(16,file='lp90mass40.dat')
        read(16,*)ms(m,n,k)
  else if(age(m,n).eq.4)then
        open(17,file='lp90ucrarea50.dat')
        read(17,*)ucrarea(m,n,k)
        open(18,file='lp90starea50.dat')
        read(18,*)starea(m,n,k)
        open(19,file='lp90mass50.dat')
        read(19,*)ms(m,n,k)
  else if(age(m,n).eq.5)then
        open(20,file='lp90ucrarea60.dat')
        read(20,*)ucrarea(m,n,k)
        open(21,file='lp90starea60.dat')
        read(21,*)starea(m,n,k)
        open(22,file='lp90mass60.dat')
        read(22,*)ms(m,n,k)
  else if(age(m,n).eq.6)then
        open(23,file='lp90ucrarea70.dat')
        read(23,*)ucrarea(m,n,k)
        open(24,file='lp90starea70.dat')
        read(24,*)starea(m,n,k)
        open(25,file='lp90mass70.dat')
        read(25,*)ms(m,n,k)
  else if(age(m,n).eq.7)then
        open(26,file='lp90ucrarea80.dat')
        read(26,*)ucrarea(m,n,k)
        open(27,file='lp90starea80.dat')
        read(27,*)starea(m,n,k)
        open(28,file=' lp90mass80.dat')
        read(28,*)ms(m,n,k)
  end if
else if(selection.eq.'5')then
    if(age(m,n).eq.1)then
        open(8,file='lp100ucrarea20.dat')
        read(8,*)ucrarea(m,n,k)
        open(9,file='lp100starea20.dat')
        read(9,*)starea(m,n,k)
        open(10,file='lp100mass20.dat')
        read(10,*)ms(m,n,k)
  else if(age(m,n).eq.2)then
        open(11,file='lp100ucrarea30.dat')
        read(11,*)ucrarea(m,n,k)
        open(12,file='lp100starea30.dat')
        read(12,*)starea(m,n,k)
        open(13,file='lp100mass30.dat')
        read(13,*)ms(m,n,k)
  else if(age(m,n).eq.3)then
        open(14,file='lp100ucrarea40.dat')
        read(14,*)ucrarea(m,n,k)
        open(15,file='lp100starea40.dat')
        read(15,*)starea(m,n,k)
        open(16,file='lp100mass40.dat')
        read(16,*)ms(m,n,k)
  else if(age(m,n).eq.4)then
        open(17,file='lp100ucrarea50.dat')
```

```
        read(17,*)ucrarea(m,n,k)
        open(18,file='lp100starea50.dat')
        read(18,*)starea(m,n,k)
        open(19,file='lp100mass50.dat')
        read(19,*)ms(m,n,k)
  else if(age(m,n).eq.5)then
        open(20,file='lp100ucrarea60.dat')
        read(20,*)ucrarea(m,n,k)
        open(21,file='lp100starea60.dat')
        read(21,*)starea(m,n,k)
        open(22,file='lp100mass60.dat')
        read(22,*)ms(m,n,k)
  else if(age(m,n).eq.6)then
        open(23,file='lp100ucrarea70.dat')
        read(23,*)ucrarea(m,n,k)
        open(24,file='lp100starea70.dat')
        read(24,*)starea(m,n,k)
        open(25,file='lp100mass70.dat')
        read(25,*)ms(m,n,k)
  else if(age(m,n).eq.7)then
        open(26,file='lp100ucrarea80.dat')
        read(26,*)ucrarea(m,n,k)
        open(27,file='lp100starea80.dat')
        read(27,*)starea(m,n,k)
        open(28,file='lp100mass80.dat')
        read(28,*)ms(m,n,k)
  end if
else if(selection.eq.'6')then
    if(age(m,n).eq.1)then
        open(8,file='lp110ucrarea20.dat')
        read(8,*)ucrarea(m,n,k)
        open(9,file='lp110starea20.dat')
        read(9,*)starea(m,n,k)
        open(10,file='lp110mass20.dat')
        read(10,*)ms(m,n,k)
  else if(age(m,n).eq.2)then
        open(11,file='lp110ucrarea30.dat')
        read(11,*)ucrarea(m,n,k)
        open(12,file='lp110starea30.dat')
        read(12,*)starea(m,n,k)
        open(13,file='lp110mass30.dat')
        read(13,*)ms(m,n,k)
  else if(age(m,n).eq.3)then
        open(14,file='lp110ucrarea40.dat')
        read(14,*)ucrarea(m,n,k)
        open(15,file='lp110starea40.dat')
        read(15,*)starea(m,n,k)
        open(16,file='lp110mass40.dat')
        read(16,*)ms(m,n,k)
  else if(age(m,n).eq.4)then
        open(17,file='lp110ucrarea50.dat')
        read(17,*)ucrarea(m,n,k)
        open(18,file='lp110starea50.dat')
        read(18,*)starea(m,n,k)
        open(19,file='lp110mass50.dat')
        read(19,*)ms(m,n,k)
  else if(age(m,n).eq.5)then
```

```
            open(20,file='lp110ucrarea60.dat')
            read(20,*)ucrarea(m,n,k)
            open(21,file='lp110starea60.dat')
            read(21,*)starea(m,n,k)
            open(22,file='lp110mass60.dat')
            read(22,*)ms(m,n,k)
      else if(age(m,n).eq.6)then
            open(23,file='lp110ucrarea70.dat')
            read(23,*)ucrarea(m,n,k)
            open(24,file='lp110starea70.dat')
            read(24,*)starea(m,n,k)
            open(25,file='lp110mass70.dat')
            read(25,*)ms(m,n,k)
      else if(age(m,n).eq.7)then
            open(26,file='lp110ucrarea80.dat')
            read(26,*)ucrarea(m,n,k)
            open(27,file='lp110starea80.dat')
            read(27,*)starea(m,n,k)
            open(28,file='lp110mass80.dat')
            read(28,*)ms(m,n,k)
      end if
    else if(selection.eq.'7')then
        if(age(m,n).eq.1)then
            open(8,file='lp120ucrarea20.dat')
            read(8,*)ucrarea(m,n,k)
            open(9,file='lp120starea20.dat')
            read(9,*)starea(m,n,k)
            open(10,file='lp120mass20.dat')
            read(10,*)ms(m,n,k)
      else if(age(m,n).eq.2)then
            open(11,file='lp120ucrarea30.dat')
            read(11,*)ucrarea(m,n,k)
            open(12,file='lp120starea30.dat')
            read(12,*)starea(m,n,k)
            open(13,file='lp120mass30.dat')
            read(13,*)ms(m,n,k)
      else if(age(m,n).eq.3)then
            open(14,file='lp120ucrarea40.dat')
            read(14,*)ucrarea(m,n,k)
            open(15,file='lp120starea40.dat')
            read(15,*)starea(m,n,k)
            open(16,file='lp120mass40.dat')
            read(16,*)ms(m,n,k)
      else if(age(m,n).eq.4)then
            open(17,file='lp120ucrarea50.dat')
            read(17,*)ucrarea(m,n,k)
            open(18,file='lp120starea50.dat')
            read(18,*)starea(m,n,k)
            open(19,file='lp120mass50.dat')
            read(19,*)ms(m,n,k)
      else if(age(m,n).eq.5)then
            open(20,file='lp120ucrarea60.dat')
            read(20,*)ucrarea(m,n,k)
            open(21,file='lp120starea60.dat')
            read(21,*)starea(m,n,k)
            open(22,file='lp120mass60.dat')
            read(22,*)ms(m,n,k)
```

```
          else if(age(m,n).eq.6)then
                open(23,file='lp120ucrarea70.dat')
                read(23,*)ucrarea(m,n,k)
                open(24,file='lp120starea70.dat')
                read(24,*)starea(m,n,k)
                open(25,file='lp120mass70.dat')
                read(25,*)ms(m,n,k)
          else if(age(m,n).eq.7)then
                open(26,file='lp120ucrarea80.dat')
                read(26,*)ucrarea(m,n,k)
                open(27,file='lp120starea80.dat')
                read(27,*)starea(m,n,k)
                open(28,file='lp120mass80.dat')
                read(28,*)ms(m,n,k)
          end if
          else if(selection.eq.'8')then
                open(8,file='lpucrarea.dat')
                read(8,*)ucrarea(m,n,k)
                open(9,file='lpstarea.dat')
                read(9,*)starea(m,n,k)
                open(10,file='lpmass.dat')
                read(10,*)ms(m,n,k)
          end if
  477 continue
          close(8)
          close(9)
          close(10)
          close(11)
          close(12)
          close(13)
          close(14)
          close(15)
          close(16)
          close(17)
          close(18)
          close(19)
          close(20)
          close(21)
          close(22)
          close(23)
          close(24)
          close(25)
          close(26)
          close(27)
          close(28)
  475 continue

      write(*,*)"Printing plot of fallen trees...(trees.ps)"

*       Print out graph of fallen trees.

        call newdev('trees.ps',8)
        call psinit(.true.)
        call factor(0.6)
        call plot(4.,120.,-3)
        call border(300.,300.,0000,1111,4,2,3,4)
        call keknum(-1.,150.,5.,200.,90.,-1,1)
```

```
      call keknum(-1.,300.,5.,400.,90.,-1,1)
      call keknum(-1.,0.,5.,0.,90.,-1,1)
      call keknum(0.,-8.,5.,-200.,0.,-1,1)
      call keknum(150.,-8.,5.,0.,0.,-1,1)
      call keknum(300.,-8.,5.,200.,0.,-1,1)
      call keksym(150.,-20.,7.,12hDistance (m),0.,12,1)
      call keksym(-8.,150.,7.,12hDistance (m),90.,12,1)
      call setlw(1.0)
      call arrow(150.,305.,150.,330.,5.,30.,0)
      call setlw(0.)
      if(answer.eq.'1')then
      call keksym(160.,320.,6.,7hTornado,0.,7,0)
      call keksym(160.,310.,6.,8hMovement,0.,8,0)
      else
      call keksym(160.,320.,6.,12hThunderstorm,0.,12,0)
      call keksym(160.,310.,6.,8hMovement,0.,8,0)
      end if
      call plot(0.,0.,-3)
      call factor(0.45)
      i=1
      n=j
      stand=0
      fallen=0
510   do 495 m=yg+1,1,-d
      do 495 j=1,xg+1,d
      call setgry(0.5)
      call circle(xss(m,j),yss(m,j),0.5,.true.)
      call setgry(0.0)
        tmax(m,j)=0.0
        fwindtot(m,j)=0.0

*     Calculate the total force due to the wind.

      do 497 o=0,h(m,j)-1
          z(o)=0.5+o

*     Calculate wind profile (vtree).

        vtree(i,j,o)=vtorn(i,j)*(log(z(o)/(0.06*h(m,j)))/
     $        log(h(m,j)/(0.06*h(m,j))))
        if(vtree(i,j,o).lt.0.)then
          vtree(i,j,o)=0.
        end if

*     Streamline the crown area and add it to the  stem area to obtain
*     total area distribution.

        if(vtree(i,j,o).le.11.0)then
            crarea(m,j,o)=ucrarea(m,j,o)*0.80
            area(m,j,o)=crarea(m,j,o)+starea(m,j,o)
        else if(vtree(i,j,o).ge.20.0)then
            crarea(m,j,o)=ucrarea(m,j,o)*0.40
            area(m,j,o)=crarea(m,j,o)+starea(m,j,o)
        else if(vtree(i,j,o).gt.11.0 .and. vtree(i,j,o).lt.20.0)then
            st(m,j,o)=(10/vtree(i,j,o))-0.1
            crarea(m,j,o)=ucrarea(m,j,o)*st(m,j,o)
            area(m,j,o)=crarea(m,j,o)+starea(m,j,o)
```

```
        end if
          fwind(m,j,o)=(0.5*cd*rho*area(m,j,o)*(vtree(i,j,o)**2))
          fwindtot(m,j)=fwindtot(m,j)+fwind(m,j,o)
  497   continue

        do 498 k=0,h(m,j)-1
          z(k)=0.5+k

*     Calculate wind profile (vtree).

        vtree(i,j,k)=vtorn(i,j)*(log(z(k)/(0.06*h(m,j)))/
     $        log(h(m,j)/(0.06*h(m,j))))
        if(vtree(i,j,k).lt.0.)then
          vtree(i,j,k)=0.
        end if

        ltop(k)=h(m,j)-z(k)

*     Streamline the crown area and add it to the stem area to obtain
*     total area distribution.

          if(vtree(i,j,k).le.11.0)then
            crarea(m,j,k)=ucrarea(m,j,k)*0.80
            area(m,j,k)=crarea(m,j,k)+starea(m,j,k)
          else if(vtree(i,j,k).ge.20.0)then
            crarea(m,j,k)=ucrarea(m,j,k)*0.40
            area(m,j,k)=crarea(m,j,k)+starea(m,j,k)
          else if(vtree(i,j,k).gt.11.0 .and. vtree(i,j,k).lt.20.0)then
            st(m,j,k)=(10/vtree(i,j,k))-0.1
            crarea(m,j,k)=ucrarea(m,j,k)*st(m,j,k)
            area(m,j,k)=crarea(m,j,k)+starea(m,j,k)
          end if
        fwind(m,j,k)=(0.5*cd*rho*area(m,j,k)*(vtree(i,j,k)**2))
        fmass(m,j,k)=g*ms(m,j,k)
        if(z(k) .ge.a(m,j))then
        horizd(m,j,k)=(fwindtot(m,j)*(a(m,j)**2)*h(m,j)*(3-(a(m,j)
     $  /h(m,j))-(3*ltop(k)/h(m,j))))/(6*moe*ami(m,j))
        else if(z(k) .lt. a(m,j))then
        horizd(m,j,k)=(fwindtot(m,j)*(a(m,j)**3)*(2-3*((ltop(k)-b(m,j))
     $  /a(m,j))+(((ltop(k)-b(m,j))**3)/a(m,j)**3)))/(6*moe*ami(m,j))
        end if
        tmom(m,j,k)=gustfac(m,j)*gapfac(m,j)*(fwind(m,j,k)*z(k)
     $          +fmass(m,j,k)*horizd(m,j,k))
        tmax(m,j)=tmax(m,j)+tmom(m,j,k)
        tmax(m,j)=tmax(m,j)-tmom(m,j,0)
  498   continue

*     Compare turning moment with the resistance to stem breakage.
*     If tmax is greater than or equal to stemres the tree is downed
*     and an arrow is plotted at that grid point in the direction of the     *
wind.
*     If the tree is downed the stemres for that tree is set equal to
*     1X10^10 so that it can not be downed again.

        if (tmax(m,j) .ge. stemres(m,j)) then
                        damage(m,j)=1
                        stemres(m,j)=1E10
```

```
                    uc(i,j)=h(m,j)*cosd(psi(i,j))
                    vc(i,j)=h(m,j)*sind(psi(i,j))
                    xpp(m,j)=xss(m,j)+uc(i,j)
                    ypp(m,j)=yss(m,j)+vc(i,j)
                    call arrow(xss(m,j),yss(m,j),xpp(m,j),
     $              ypp(m,j),3.,30.,0)
        end if

        if(damage(m,j).eq.0.or.damage(m,j).eq.1)then
            stand=stand+1
        end if

        if(damage(m,j).eq.1)then
            fallen=fallen+1
        end if

 495    continue
        i=i+d
        if (i .le. yg+1) then
                goto 510
        end if

        percent=(fallen/stand)*100

*     Take the height and dbh values out of array format so they can be
*     displayed on the plot.

      ht=-1.e20
      db=-1.e20
      do 600 m=1,yg+1
      do 600 n=1,xg+1
            ht=amax1(ht,h(m,n))
              db=amax1(db,dbh(m,n))
 600  continue
      db=db*100


      call factor(0.45)
      call keksym(10.,-40.,7.,13hLoblolly Pine,0.,13,0)
      if(selection.eq.'1')then
      call keksym(999.,999.,7.,18h     Site Index=60,0.,18,0)
      else if(selection.eq.'2')then
      call keksym(999.,999.,7.,18h     Site Index=70,0.,18,0)
      else if(selection.eq.'3')then
      call keksym(999.,999.,7.,18h     Site Index=80,0.,18,0)
      else if(selection.eq.'4')then
      call keksym(999.,999.,7.,18h     Site Index=90,0.,18,0)
      else if(selection.eq.'5')then
      call keksym(999.,999.,7.,19h     Site Index=100, 0.,19,0)
      else if(selection.eq.'6')then
      call keksym(999.,999.,7.,19h     Site Index=110,0.,19,0)
      else if(selection.eq.'7')then
      call keksym(999.,999.,7.,19h     Site Index=120,0.,19,0)
      else if(selection.eq.'8')then
      call keksym(999.,999.,7.,5h   h=,0.,5,0)
      call keknum(999.,999.,7.,ht,0.,1,0)
      call keksym(999.,999.,7.,1hm,0.,1,0)
```

```
 call keksym(999.,999.,7.,6h  dbh=,0.,6,0)
 call keknum(999.,999.,7.,db,0.,1,0)
 call keksym(999.,999.,7.,2hcm,0.,2,0)
 call keksym(999.,999.,7.,13h     spacing=,0.,13,0)
 call keknum(999.,999.,7.,grid,0.,1,0)
 call keksym(999.,999.,7.,1hm,0.,1,0)
 end if
 if(answer.eq.'1')then
 call keksym(10.,-50.,7.,27hMax. tangential velocity = ,0.,
$      27,0)
 call keknum(999.,999.,7.,vmax,0.,-1,0)
 call keksym(999.,999.,7.,4h m/s,0.,4,0)
 call keksym(10.,-60.,7.,23hRadius of max. winds = ,0.,23,0)
 call keknum(999.,999.,7.,rc,0.,-1,0)
 call keksym(999.,999.,7.,2h m,0.,2,0)
 call keksym(10.,-70.,7.,23hMax. radial velocity = ,0.,23,0)
 call keknum(999.,999.,7.,vrmax,0.,-1,0)
 call keksym(999.,999.,7.,4h m/s,0.,4,0)
 call keksym(10.,-80.,7.,16hForward speed = ,0.,16,0)
 call keknum(999.,999.,7.,vforw,0.,-1,0)
 call keksym(999.,999.,7.,4h m/s,0.,4,0)
 call keksym(10.,-90.,7.,21hMaximum wind speed = ,0.,21,0)
 call keknum(999.,999.,7.,vtornmax,0.,-1,0)
 call keksym(999.,999.,7.,7h m/s = ,0.,7,0)
 call keknum(999.,999.,7.,vtornmaxmph,0.,-1,0)
 call keksym(999.,999.,7.,4h mph,0.,4,0)
 else
 call keksym(10.,-50.,7.,27hMax. microburst velocity = ,0.,
$      27,0)
 micromax=-vrmax
 call keknum(999.,999.,7.,micromax,0.,-1,0)
 call keksym(999.,999.,7.,4h m/s,0.,4,0)
 call keksym(10.,-60.,7.,23hWidth of damage path = ,0.,23,0)
 call keknum(999.,999.,7.,rc,0.,-1,0)
 call keksym(999.,999.,7.,2h m,0.,2,0)
 call keksym(10.,-70.,7.,16hForward speed = ,0.,16,0)
 call keknum(999.,999.,7.,vforw,0.,-1,0)
 call keksym(999.,999.,7.,4h m/s,0.,4,0)
 call keksym(10.,-80.,7.,21hMaximum wind speed = ,0.,21,0)
 call keknum(999.,999.,7.,vtornmax,0.,-1,0)
 call keksym(999.,999.,7.,7h m/s = ,0.,7,0)
 call keknum(999.,999.,7.,vtornmaxmph,0.,-1,0)
 call keksym(999.,999.,7.,4h mph,0.,4,0)
 end if
 if(answer.eq.'1')then
 if((vtornmaxmph.ge.40.).and.(vtornmaxmph.le.72.))then
 call keksym(10.,-100.,7.,10hF0 Tornado,0.,10,0)
 else if((vtornmaxmph.ge.73.).and.(vtornmaxmph.le.112.))then
 call keksym(10.,-100.,7.,10hF1 Tornado,0.,10,0)
 else if((vtornmaxmph.ge.113.).and.(vtornmaxmph.le.157.))then
 call keksym(10.,-100.,7.,10hF2 Tornado,0.,10,0)
 else if((vtornmaxmph.ge.158.).and.(vtornmaxmph.le.206.))then
 call keksym(10.,-100.,7.,10hF3 Tornado,0.,10,0)
 else if((vtornmaxmph.ge.207.).and.(vtornmaxmph.le.260.))then
 call keksym(10.,-100.,7.,10hF4 Tornado,0.,10,0)
 else if((vtornmaxmph.ge.261.).and.(vtornmaxmph.le.318.))then
 call keksym(10.,-100.,7.,10hF5 Tornado,0.,10,0)
```

```
      else if(vtornmaxmph.ge.319)then
      call keksym(10.,-100.,7.,10hF6 Tornado,0.,10,0)
      end if
      call keksym(10.,-110.,7.,37hSize of upwind gap in tree height
     $      s = ,0.,37,0)
      call keknum(999.,999.,7.,gapsize,0.,-1,0)
      call keksym(10.,-120.,7.,25hEvery other tree plotted.,0.,25,0)
      call keksym(10.,-130.,7.,26hPercent of trees felled = ,0.,26,0)
      call keknum(999.,999.,7.,percent,0.,1,0)
      call keksym(999.,999.,7.,1h%,0.,1,0)
      else if(answer.eq.'2')then
      if((vtornmaxmph.ge.40.).and.(vtornmaxmph.le.72.) )then
      call keksym(10.,-90.,7.,13hF0 Microburst,0.,13,0)
      else if((vtornmaxmph.ge.73.).and.(vtornmaxmph.le.112.))then
      call keksym(10.,-90.,7.,13hF1 Microburst,0.,13,0)
      else if((vtornmaxmph.ge.113.).and.(vtornmaxmph.le.157.))then
      call keksym(10.,-90.,7.,13hF2 Microburst,0.,13,0)
      else if((vtornmaxmph.ge.158.).and.(vtornmaxmph.le.206.))then
      call keksym(10.,-90.,7.,13hF3 Microburst,0.,13,0)
      else if((vtornmaxmph.ge.207.).and.(vtornmaxmph.le.260.))then
      call keksym(10.,-90.,7.,13hF4 Microburst,0.,13,0)
      else if((vtornmaxmph.ge.261.).and.(vtornmaxmph.le.318.))then
      call keksym(10.,-90.,7.,13hF5 Microburst,0.,13,0)
      else if(vtornmaxmph.ge.319)then
      call keksym(10.,-90.,7.,13hF6 Microburst,0.,13,0)
      end if
      call keksym(10.,-100.,7.,37hSize of upwind gap in tree height
     $      s = ,0.,37,0)
      call keknum(999.,999.,7.,gapsize,0.,-1,0)
      call keksym(10.,-110.,7.,25hEvery other tree plotted.,0.,25,0)
      call keksym(10.,-120.,7.,26hPercent of trees felled = ,0.,26,0)
      call keknum(999.,999.,7.,percent,0.,1,0)
      call keksym(999.,999.,7.,1h%,0.,1,0)
      end if
      call keksym(275.,-125.,5.,8hTornTree,0.,8,0)

      call plotnd

      return
      end


**SCOTSPINE*************************************************************
*                                                                      *
*  Calculates the turning moment and generates a graphical output file,*
*  'trees.ps.'                                                         *
*                                                                      *
************************************************************************

      subroutine scotspine(grid,d,ran,vtorn,ucom,vcom,gap size,
    $   selection,vmax,rc,vrmax,vforw,vtornmax,vtornmaxmph,answer,psi)

      real x(300,0:300),y(300,300)    !(x,y) coordinates of each point
      real xss(300,300),yss(300,300)  !starting point of the arrow
      real xpp(300,300),ypp(300,300)  ! end point of the arrow
      real z(0:50)                    !height of each one -meter segment
                                           !(m)
```

```
real vtree(300,300,0:50)        !3-D wind field of vortex (m/s)
real ran(300,300)               !random number
real age(300,300)               !randomly determined age class of
                                     !tree
real rownum(300,300)            !number of row in the forest
real distedge(300,300)          !distance from forest edge in tree
                                     !heights
real gustmax(300,300)           !used to calculate gust factor
real gustmeanedge(300,300)      !used to calculate gust factor
real gustfac(300,300)           !factor to include gustiness of
                                     !wind
real gapmax(300,300)            !used to calculate gap factor
real gapmeanedge(300,300)       !used to calculate gap factor
real gapfac(300,300)            !factor to include size of the
                                     !upwind gap
real stemres(300,300)           !resistance to stem breakage (Nm)
real area(300,300,0:50)         !area of each one-meter segment
                                     !(m^2)
real ms(300,300,0:50)           !mass of each one-meter segment
                                     !(kg)
real fwind(300,300,0:50)        !force due to the wind (N)
real fwindtot(300,300)          !sum of fwind over the tree height
                                     !(N)
real fmass(300,300,0:50)        !force due to gravity (N)
real ltop(0:50)                 !distance from tree top (m)
real horizd(300,300,0:50)       !horizontal displacement of the
                                     !stem (m)
real h(300,300)                 !height of the tree (m)
real cd                         !drag coefficient
real rho                        !density of air (kg/m^3)
real g                          !gravitational constant (m/s^2)
real dbh(300,300)               !diameter at breast height (m)
real a(300,300)                 !distance from ground to crown
                                     !center (m)
real b(300,300)                 !distance from tree top to crown
                                     !center (m)
real crhgt(300,300)             !height from ground to crown base
                                     !(m)
real stemhgt(300,300)           !total height - crhgt, stem height
                                     !(m)
real moe                        !modulus of elasticity (Pa)
real mor                        !modulus of rupture (Pa)
real ami(300,300)               !area moment of inertia (m^4)
real gh(300,300)                !spacing between trees in tree
                                     !heights
real grid                       !tree spacing (m)
real xg,yg                      !number of points in the grid
real xmax,xmin,ymax,ymin        !max and min values of x and y
real gapsize                    !size of the upwind gap in tree
                                     !heights
real d                          !counter to determine number of
                                     !trees drawn
real ucrarea(300,300,0:50)      !unstreamlined crown area (m^2)
real starea(300,300,0:50)       !stem area (m^2)
real crarea(300,300,0:50)       !streamlined crown area (m^2)
real st(300,300,0:50)           !streamlining function
real ucom(300,300),vcom(300,300) !forward motion plus utran and
```

```
                                              !vtran (m/s)
      real vtorn(300,300)               !vector of vortex winds (m/s)
      real vtornmax                     !max vortex wind speed
      real ht,db                        !height and dbh
      character selection               !the menu selection
      character answer                  !choice between tornado and
                                             !microburst
      real tmax(300,300)                !maximum turning moment (Nm)
      real tmom(300,300,0:50)           !turning moment for segment (Nm)
      real micromax                     !max. microburst wind speed
      real psi(300,300)                 !angle of vtorn from the east
      real uc(300,300)                  !used to make the length of the
                                             !fallen tree vector
                                             !proportional to tree height
      real vc(300,300)                  !"
      integer damage(300,300)           !0 for standing tree, 1 for fallen tree
      real stand                        !number of trees in the stand
      real fallen                       !number of fallen trees
      real percent                      !percent of fallen trees

      xmax=200
      xmin=-200
      ymax=200
      ymin=-200
      pi=3.14
      cd=0.29
      rho=1.226
      g=9.8
      moe=7E9
      mor=39100000.

      xg=xmax*2/grid
      yg=ymax*2/grid

*       Calculate x and y for the tree grid.

        do 400 m=1,yg+1
        do 400 n=0
                x(m,n)=xmin-grid
  400   continue

        do 410 m=1,yg+1
        do 410 n=1,xg+1
                x(m,n)=x(m,n-1)+grid
  410   continue

        do 420 m=1,yg+1
        do 420 n=1,xg+1
                if (m .eq. 1) then
                        y(m,n)=ymax
                else if (m .ne. 1) then
                        y(m,n)=y(m-1,n)-grid
                end if
  420   continue

      do 425 m=1,yg+1
      do 425 n=1,xg+1
```

```
            xss(m,n)=x(m,n)+200
            yss(m,n)=y(m,n)+200
  425 continue

*       Randomly pick age class of each tree.

        do 430 m=1,yg+1
        do 430 n=1,xg+1
             call random(ran(m,n))
               if (ran(m,n) .ge. 0 .and. ran(m,n) .le. 0.2)then
                                age(m,n)=1
               else if(ran(m,n).gt.0.2 .and. ran(m,n).le.0.5)then
                                age(m,n)=2
               else if(ran(m,n).gt.0.5 .and. ran(m,n).le.0.8)then
                                age(m,n)=3
               else if(ran(m,n).gt.0.8 .and. ran(m,n).le.1.0)then
                                age(m,n)=4
               end if
  430   continue

*       Assign each row of the forest a number.

        do 456 m=1,yg+1
        do 456 n=1,xg+1
                rownum(m,n)=yg+1-m
  456   continue

*       Read in values based on menu choice.
*       'spage*.dat' -> Tree height and dbh values for * age class.

        do 457 m=1,yg+1
        do 457 n=1,xg+1
          if(selection.eq.'1')then
                open(1,file='spage35.dat')
                read(1,*)dbh(m,n),h(m,n)
                close(1)
                goto 460
          else if(selection.eq.'2')then
                open(2,file='spage65.dat')
                read(2,*)dbh(m,n),h(m,n)
                close(2)
                goto 460
          else if(selection.eq.'3')then
                open(3,file='spage95.dat')
                read(3,*)dbh(m,n),h(m,n)
                close(3)
                goto 460
          else if(selection.eq.'4')then
                open(4,file='spage120.dat')
                read(4,*)dbh(m,n),h(m,n)
                close(4)
                goto 460
          else if(selection.eq.'5')then
                goto 459
          end if
  459     if(age(m,n).eq.1)then
                open(1,file='spage35.dat')
```

```
                read(1,*)dbh(m,n),h(m,n)
                close(1)
                goto 460
          else if(age(m,n).eq.2)then
                open(2,file='spage65.dat')
                read(2,*)dbh(m,n),h(m,n)
                close(2)
                goto 460
          else if(age(m,n).eq.3)then
                open(3,file='spage95.dat')
                read(3,*)dbh(m,n),h(m,n)
                close(3)
                goto 460
          else if(age(m,n).eq.4)then
                open(4,file='spage120.dat')
                read(4,*)dbh(m,n),h(m,n)
                close(4)
                goto 460
          end if
  460   gh(m,n)=grid/h(m,n)
        ami(m,n)=pi*((dbh(m,n)**4)/64)
        crhgt(m,n)=(h(m,n)*0.42)
        stemhgt(m,n)=h(m,n)-crhgt(m,n)
        a(m,n)=(crhgt(m,n)/2)+stemhgt(m,n)
        b(m,n)=h(m,n)-a(m,n)
  457   continue


*     Calulate the gust and gap factors and the tree's resistance to
*     stem breakage.

        do 470 m=yg+1,1,-1
        do 470 n=1,xg+1
        distedge(m,n)=gh(m,n)*rownum(m,n)
        gustmeanedge(m,n)=(((1.7239*gh(m,n)+0.0316)**(0.))
     $  *(-0.68*gh(m,n)+0.4785))+(0.68*gh(m,n)-0.0385)
        gustmax(m,n)=(((1.1127*gh(m,n)+0.0311)**(distedge(m,n)))
     $  *(-1.273*gh(m,n)+0.9701))+(2.7193*gh(m,n)-0.061)
        gustfac(m,n)=gustmax(m,n)/gustmeanedge(m,n)
        gapmeanedge(m,n)=(0.001+0.001*(10.**0.562))/0.00465
        gapmax(m,n)=(0.0072+0.0064*(gapsize**0.3467))/0.0214
        gapfac(m,n)=gapmax(m,n)/gapmeanedge(m,n)
        stemres(m,n)=(pi/32)*mor*(dbh(m,n)**3)
  470   continue

*     Read in the area and mass distributions of each tree.
*     'spucrarea*.dat' -> Unstreamlined crown area distribution.
*     'spstarea*.dat' -> Stem area distribution.
*     'spmass*.dat' -> Stem and crown mass distribution.


      do 475 m=1,yg+1
      do 475 n=1,xg+1
      do 477 k=0,h(m,n)-1
        if(selection.eq.'1')then
                open(5,file='spucrarea35.dat')
                read(5,*)ucrarea(m,n,k)
```

```
            open(6,file='spstarea35.dat')
            read(6,*)starea(m,n,k)
            open(7,file='spmass35.dat')
            read(7,*)ms(m,n,k)
            goto 477
      else if(selection.eq.'2')then
            open(8,file='spucrarea65.dat')
            read(8,*)ucrarea(m,n,k)
            open(9,file='spstarea65. dat')
            read(9,*)starea(m,n,k)
            open(10,file='spmass65.dat')
            read(10,*)ms(m,n,k)
            goto 477
      else if(selection.eq.'3')then
            open(11,file='spucrarea95.dat')
            read(11,*)ucrarea(m,n,k)
            open(12,file='spstarea95.dat')
            read(12,*)starea(m,n,k)
            open(13,file='spmass95.dat')
            read(13,*)ms(m,n,k)
            goto 477
      else if(selection.eq.'4')then
            open(14,file='spucrarea120.dat')
            read(14,*)ucrarea(m,n,k)
            open(15,file='spstarea120.dat')
            read(15,*)starea(m,n,k)
            open(16,file='spmass120.dat')
            read(16,*)ms(m,n,k)
            goto 477
      else if(selection.eq.'5')then
            goto 480
      end if
480   if(age(m,n).eq.1)then
            open(5,file='spucrarea35.dat')
            read(5,*)ucrarea(m,n,k)
            open(6,file='spstarea35.dat')
            read(6,*)starea(m,n,k)
            open(7,file='spmass35.dat')
            read(7,*)ms(m,n,k)
      else if(age(m,n).eq.2)then
            open(8,file='spucrarea65.dat')
            read(8,*)ucrarea(m,n,k)
            open(9,file='spstarea65.dat')
            read(9,*)starea(m,n,k)
            open(10,file='spmass65.dat')
            read(10,*)ms(m,n,k)
      else if(age(m,n).eq.3)then
            open(11,file='spucrarea95.dat')
            read(11,*)ucrarea(m,n,k)
            open(12,file='spstarea95.dat')
            read(12,*)starea(m,n,k)
            open(13,file='spmass95.dat')
            read(13,*)ms(m,n,k)
      else if(age(m,n).eq.4)then
            open(14,file='spucrarea120.dat')
            read(14,*)ucrarea(m,n,k)
            open(15,file='spstarea120.dat')
```

```
                read(15,*)starea(m,n,k)
                open(16,file='spmass120.dat')
                read(16,*)ms(m,n,k)
          end if
  477 continue
          close(5)
          close(6)
          close(7)
          close(8)
          close(9)
          close(10)
          close(11)
          close(12)
          close(13)
          close(14)
          close(15)
          close(16)
  475 continue

*       Print out graph of fallen trees.

        call newdev('trees.ps',8)
        call psinit(.true.)
        call factor(0.6)
        call plot(4.,120.,-3)
        call border(300.,300.,0000,1111,4,2,3,4)
        call keknum(-1.,150.,5.,200.,90.,-1,1)
        call keknum(-1.,300.,5.,400.,90.,-1,1)
        call keknum(-1.,0.,5.,0.,90.,-1,1)
        call keknum(0.,-8.,5.,-200.,0.,-1,1)
        call keknum(150.,-8.,5.,0.,0.,-1,1)
        call keknum(300.,-8.,5.,200.,0.,-1,1)
        call keksym(150.,-20.,7.,12hDistance (m),0.,12,1)
        call keksym(-8.,150.,7.,12hDistance (m),90.,12,1)
        call setlw(1.0)
        call arrow(150.,305.,150.,330.,5.,30.,0)
        call setlw(0.)
        if(answer.eq.'1')then
        call keksym(160.,320.,6.,7hTornado,0.,7,0)
        call keksym(160.,310.,6.,8hMovement,0.,8,0)
        else
        call keksym(160.,320.,6.,12hThunderstorm,0.,12,0)
        call keksym(160.,310.,6.,8hMovement,0.,8,0)
        end if
        call plot(0.,0.,-3)
        call factor(0.45)
        i=1
        n=j
        stand=0
        fallen=0
  510   do 495 m=yg+1,1,-d
        do 495 j=1,xg+1,d
        call setgry(0.5)
        call circle(xss(m,j),yss(m,j),0.5,.true.)
        call setgry(0.0)
          tmax(m,j)=0.0
          fwindtot(m,j)=0.0
```

```
*     Calculate the total force due to the wind.

      do 497 o=0,h(m,j)-1
        z(o)=0.5+o

*     Calculate wind profile (vtree).

        vtree(i,j,o)=vtorn(i,j)*(log(z(o)/(0.06*h(m ,j)))/
     $        log(h(m,j)/(0.06*h(m,j))))
        if(vtree(i,j,o).lt.0.)then
          vtree(i,j,o)=0.
        end if

*     Streamline the crown area and add it to the stem area to obtain
*     total area distribution.

          if(vtree(i,j,o).le.11.0)then
            crarea(m,j,o)=ucrarea(m,j,o)*0.80
            area(m,j,o)=crarea(m,j,o)+starea(m,j,o)
        else if(vtree(i,j,o).ge.20.0)then
            crarea(m,j,o)=ucrarea(m,j,o)*0.40
            area(m,j,o)=crarea(m,j,o)+starea(m,j,o)
        else if(vtree(i,j,o).gt.11.0 .and. vtree(i,j,o).lt.20.0)then
            st(m,j,o)=(10/vtree(i,j,o))-0.1
            crarea(m,j,o)=ucrarea(m,j,o)*st(m,j,o)
            area(m,j,o)=crarea(m,j,o)+starea(m,j,o)
        end if
          fwind(m,j,o)=(0.5*cd*rho*area(m,j,o)*(vtree(i,j,o)**2))
          fwindtot(m,j)=fwindtot(m,j)+fwind(m,j,o)
  497   continue

        do 498 k=0,h(m,j)-1
        z(k)=0.5+k

*     Calculate wind profile (vtree).

        vtree(i,j,k)=vtorn(i,j)*(log(z(k)/(0.06*h(m,j)))/
     $        log(h(m,j)/(0.06*h(m,j))))
        if(vtree(i,j,k).lt.0.)then
          vtree(i,j,k)=0.
        end if

          ltop(k)=h(m,j)-z(k)


*     Streamline the crown area and add it to the stem area to obtain total
*     area distribution.

         if(vtree(i,j,k).le.11.0)then
            crarea(m,j,k)=ucrarea(m,j,k)*0.80
            area(m,j,k)=crarea(m,j,k)+starea(m,j,k)
        else if(vtree(i,j,k).ge.20.0)then
            crarea(m,j,k)=ucrarea(m,j,k)*0.40
            area(m,j,k)=crarea(m,j,k)+starea(m,j,k)
        else if(vtree(i,j,k).gt.11.0 .and. vtree(i,j,k).lt.20.0)then
            st(m,j,k)=(10/vtree(i,j,k))-0.1
```

```
            crarea(m,j,k)=ucrarea(m,j,k)*st(m,j,k)
            area(m,j,k)=crarea(m,j,k)+starea(m,j,k)
        end if
        fwind(m,j,k)=(0.5*cd*rho*area(m,j,k)*(vtree(i,j,k)**2))
        fmass(m,j,k)=g*ms(m,j,k)
        if(z(k) .ge. a(m,j))then
        horizd(m,j,k)=(fwindtot(m,j)*(a(m,j)**2)*h(m,j)*(3 -(a(m,j)
     $  /h(m,j))-(3*ltop(k)/h(m,j)))))/(6*moe*ami(m,j))
        else if(z(k) .lt. a(m,j))then
        horizd(m,j,k)=(fwindtot(m,j)*(a(m,j)**3)*(2 -3*((ltop(k)-b(m,j))
     $  /a(m,j))+(((ltop(k)-b(m,j))**3)/a(m,j)**3)))/(6*moe*ami(m,j))
        end if
        tmom(m,j,k)=gustfac(m,j)*gapfac(m,j)*(fwind(m,j,k)*z(k)
     $          +fmass(m,j,k)*horizd(m,j,k))
        tmax(m,j)=tmax(m,j)+tmom(m,j,k)
        tmax(m,j)=tmax(m,j)-tmom(m,j,0)
  498   continue

*     Compare turning moment to resistance to stem breakage.
*     If tmax is greater than or equal to stemres the tree is downed
*     and an arrow is plotted at that grid point in the direction of the
*     wind.
*     If the tree is downed the stemres for that tree is set equal to
*     1X10^10 so that it can not be downed again.

        if (tmax(m,j) .ge. stemres(m,j)) then
                        damage(m,j)=1
                        stemres(m,j)=1E10
                        uc(i,j)=h(m,j)*cosd(psi(i,j))
                        vc(i,j)=h(m,j)*sind(psi(i,j))
                        xpp(m,j)=xss(m,j)+uc(i,j)
                        ypp(m,j)=yss(m,j)+vc(i,j)
                        call arrow(xss(m,j),yss(m,j),xpp(m,j),
     $                  ypp(m,j),3.,30.,0)
        end if

        if(damage(m,j).eq.0.or.damage(m,j).eq.1)then
            stand=stand+1
        end if

        if(damage(m,j).eq.1)then
            fallen=fallen+1
        end if

  495   continue
        i=i+d
        if (i .le. yg+1) then
                goto 510
        end if

        percent=(fallen/stand)*100

*     Take the height and dbh values out of array format so they can be
*     displayed on the plot.

      ht=-1.e20
      db=-1.e20
```

```
      do 600 m=1,yg+1
      do 600 n=1,xg+1
            ht=amax1(ht,h(m,n))
              db=amax1(db,dbh(m,n))
  600 continue
      db=db*100

      call factor(0.45)
      call keksym(10.,-40.,7.,10hScots Pine,0.,10,0)
      if(selection.eq.'5')then
        call keksym(999.,999.,7.,11h     Random,0.,11,0)
      else
        call keksym(999.,999.,7.,5h   h=,0.,5,0)
        call keknum(999.,999.,7.,ht,0.,1,0)
        call keksym(999.,999.,7.,1hm,0.,1,0)
        call keksym(999.,999.,7.,6h  dbh=,0.,6,0)
        call keknum(999.,999.,7.,db,0.,1,0)
        call keksym(999.,999.,7.,2hcm,0.,2,0)
        call keksym(999.,999.,7.,13h     spacing=,0.,13,0)
        call keknum(999.,999.,7.,grid,0.,1,0)
        call keksym(999.,999.,7.,1hm,0.,1,0)
      end if
      if(answer.eq.'1')then
      call keksym(10.,-50.,7.,27hMax. tangential velocity = ,0.,
     $      27,0)
      call keknum(999.,999.,7.,vmax,0.,-1,0)
      call keksym(999.,999.,7.,4h m/s,0.,4,0)
      call keksym(10.,-60.,7.,23hRadius of max. winds = ,0.,23,0)
      call keknum(999.,999.,7.,rc,0.,-1,0)
      call keksym(999.,999.,7.,2h m,0.,2,0)
      call keksym(10.,-70.,7.,23hMax. radial velocity = ,0.,23,0)
      call keknum(999.,999.,7.,vrmax,0.,-1,0)
      call keksym(999.,999.,7.,4h m/s,0.,4,0)
      call keksym(10.,-80.,7.,16hForward speed = ,0.,16,0)
      call keknum(999.,999.,7.,vforw,0.,-1,0)
      call keksym(999.,999.,7.,4h m/s,0.,4,0)
      call keksym(10.,-90.,7.,21hMaximum wind speed = ,0.,21,0)
      call keknum(999.,999.,7.,vtornmax,0.,-1,0)
      call keksym(999.,999.,7.,7h m/s = ,0.,7,0)
      call keknum(999.,999.,7.,vtornmaxmph,0.,-1,0)
      call keksym(999.,999.,7.,4h mph,0.,4,0)
      else
      call keksym(10.,-50.,7.,21hMaximum wind speed = ,0.,
     $      21,0)
      micromax=-vrmax
      call keknum(999.,999.,7.,micromax,0.,-1,0)
      call keksym(999.,999.,7.,4h m/s,0.,4,0)
      call keksym(10.,-60.,7.,23hWidth of damage path = ,0.,23,0)
      call keknum(999.,999.,7.,rc,0.,-1,0)
      call keksym(999.,999.,7.,2h m,0.,2,0)
      call keksym(10.,-70.,7.,16hForward speed = ,0.,16,0)
      call keknum(999.,999.,7.,vforw,0.,-1,0)
      call keksym(999.,999.,7.,4h m/s,0.,4,0)
      call keksym(10.,-80.,7.,21hMaximum wind speed = ,0.,21,0)
      call keknum(999.,999.,7.,vtornmax,0.,-1,0)
      call keksym(999.,999.,7.,7h m/s = ,0.,7,0)
      call keknum(999.,999.,7.,vtornmaxmph,0.,-1,0)
```

```
 call keksym(999.,999.,7.,4h mph,0.,4,0)
 end if
 if(answer.eq.'1')then
 if((vtornmaxmph.ge.40.).and.(vtornmaxmph.le.72.))then
 call keksym(10.,-100.,7.,10hF0 Tornado,0.,10,0)
 else if((vtornmaxmph.ge.73.).and.(vtornmaxmph.le.112.))then
 call keksym(10.,-100.,7.,10hF1 Tornado,0.,10,0)
 else if((vtornmaxmph.ge.113.).and.(vtornmaxmph.le.157.))then
 call keksym(10.,-100.,7.,10hF2 Tornado,0.,10,0)
 else if((vtornmaxmph.ge.158.).and.(vtornmaxmph.le.206.))then
 call keksym(10.,-100.,7.,10hF3 Tornado,0.,10,0)
 else if((vtornmaxmph.ge.207.).and.(vtornmaxmph.le.260.))then
 call keksym(10.,-100.,7.,10hF4 Tornado,0.,10,0)
 else if((vtornmaxmph.ge.261.).and.(vtornmaxmph.le.318.))then
 call keksym(10.,-100.,7.,10hF5 Tornado,0.,10,0)
 else if(vtornmaxmph.ge.319)then
 call keksym(10.,-100.,7.,10hF6 Tornado,0.,10,0)
 end if
 call keksym(10.,-110.,7.,37hSize of upwind gap in tree height
$      s = ,0.,37,0)
 call keknum(999.,999.,7.,gapsize,0.,-1,0)
 call keksym(10.,-130.,7.,26hPercent of trees felled = ,0.,26,0)
 call keknum(999.,999.,7.,percent,0.,1,0)
 call keksym(999.,999.,7.,1h%,0.,1,0)
 if(selection.eq.'1')then
  call keksym(10.,-120.,7.,26hEvery fourth tree plotted.,0.,26,0)
 else if(selection.eq.'2')then
  call keksym(10.,-120.,7.,25hEvery third tree plotted.,0.,25,0)
 else
  call keksym(10.,-120.,7.,25hEvery other tree plotted.,0.,25,0)
 end if
 else if(answer.eq.'2')then
 if((vtornmaxmph.ge.40.).and.(vtornmaxmph.le.72.))then
 call keksym(10.,-90.,7.,13hF0 Microburst,0.,13,0)
 else if((vtornmaxmph.ge.73.).and.(vtornmaxmph.le.112.))then
 call keksym(10.,-90.,7.,13hF1 Microburst,0.,13,0)
 else if((vtornmaxmph.ge.113.).and.(vtornmaxmph.le.157.))then
 call keksym(10.,-90.,7.,13hF2 Microburst,0.,13,0)
 else if((vtornmaxmph.ge.158.).and.(vtornmaxmph.le.206.))then
 call keksym(10.,-90.,7.,13hF3 Microburst,0.,13,0)
 else if((vtornmaxmph.ge.207.).and.(vtornmaxmph.le.260.))then
 call keksym(10.,-90.,7.,13hF4 Microburst,0.,13,0)
 else if((vtornmaxmph.ge.261.).and.(vtornmaxmph.le.318.))then
 call keksym(10.,-90.,7.,13hF5 Microburst,0.,13,0)
 else if(vtornmaxmph.ge.319)then
 call keksym(10.,-90.,7.,13hF6 Microburst,0.,13,0)
 end if
 call keksym(10.,-100.,7.,37hSize of upwind gap in tree height
$      s = ,0.,37,0)
 call keknum(999.,999.,7.,gapsize,0.,-1,0)
 call keksym(10.,-120.,7.,26hPercent of trees felled = ,0.,26,0)
 call keknum(999.,999.,7.,percent,0.,1,0)
 call keksym(999.,999.,7.,1h%,0.,1,0)
 if(selection.eq.'1')then
  call keksym(10.,-110.,7.,26hEvery fourth tree plotted.,0.,26,0)
 else if(selection.eq.'2')then
  call keksym(10.,-110.,7.,25hEvery third tree plotted.,0.,25,0)
```

```
      else
       call keksym(10.,-110.,7.,25hEvery other tree plotted.,0.,25,0)
      end if
      end if
      call keksym(275.,-125.,5.,8hTornTree,0.,8,0)

      call plotnd

      return
      end



******RANDOM****************************************************
*     A random number generator.                               *
****************************************************************

        subroutine random(rannum)
        integer m, const1
        real rannum, const2
        parameter (const1=2147483647, const2=.4656613E -9)
        save
        data m /0/

        if (m .eq. 0) m=int(rannum)
        m=m*65539
        if (m .lt. 0) m=(m+1)+const1
        rannum=m*const2
        return
        end


*****CLEARSCREEN************************************************
*    Clears the screen.                                        *
****************************************************************

        subroutine clrscrn

        write(*,1)char(27),'[;h',char(27),'[2j'
  1     format(1x,4a)
        return
        end

*****COLORPLOT*************************************************************
*                                                                         *
*     Creates a color contour plot of the vortex winds, 'contour.ps'.  *     *
*
***************************************************************************

      subroutine colorplot(d,grid,ucom,vcom,vmax,rc,
     $      vrmax,vforw,vtornmax,vtornmaxmph,xss,yss,answer)

      real xc(300,0:300),yc(300,300)    !(x,y) coordinates of each point
      real rcc(300,300)                 !radius of each point in meters
      real vtanc(300,300)               !tangential velocity at each
                                          !point (m/s)
      real thetac(300,300),gammac(300,300) !angle of the tangential wind
```

```
                                            !measured from the x-axis
      real xss(300,300),yss(300,300)    !starting point of the arrow
      real xpp(300,300),ypp(300,300)    !end point of the arrow
      real utc(300,300),vtc(300,300)    !vector components of tangential
                                          !wind(m/s)
      real utac(300,300),vtac(300,300)  !ut and vt times 0.3
                                          !(scaling factor) for plots
      real vradc(300,300)               !radial velocity in m/s
      real deltac(300,300)              !angle of the radial wind
                                          !measured from the x-axis
      real urc(300,300),vrc(300,300)    !vector components of the
                                            !radial wind (m/s)
      real urac(300,300),vrac(300,300)  !ur and vr times 0.3
      real utrc(300,300),vtrc(300,300)  !tangential plus radial wind
                                          !vectors
      real vtranc(300,300),utranc(300,300) !vector components of forward
                                                !motion(m/s)
      real ucom(300,300),vcom(300,300)  !forward motion plus utran and
                                            !vtran
      real ucomc(300,300),vcomc(300,300)!"
      real vtornc(41,41)                !vector of vortex winds in m/s
      real rc                           !radius of the max winds in
                                           !meters
      real vmax                         !maximum tangential velocity in
                                           !m/s
      real grid                         !tree spacing in meters
      real xg,yg                        !number of points in the grid
      real xmax,xmin,ymax,ymin          !max and min values of x and y
      real vforw                        !forward speed of the vortex in
                                           !m/s
      real vrmax                        !maximum radial wind speed in
                                           !m/s
      real d                            !counter to determine number of
                                           !trees drawn
      real vtornmax                   !maximum vortex wind speed in m/s
      real vtornmaxmph                !max vortex wind speed in mph
      real micromax                   !max. microburst wind speed
      real gridc                      !spacing between the trees in
                                           !meters
      real cval(100)                  !array containing values to be
                                           !used for contour intervals
      integer nval                    !number of contour intervals
      real color(3,100)               !array containing red, green and
                                           !blue values
      character answer                !choice between tornado and
                                           !microburst

      xmax=200
      xmin=-200
      ymax=200
      ymin=-200
      pi=3.14

      gridc=10
      xg=xmax*2/gridc

*       Calculate y.
```

```
      do 5 i=1,xg+1
      do 6 j=0
              yc(i,j)=ymin-gridc
  6   continue
  5   continue

      do 7 i=1,xg+1
      do 8 j=1,xg+1
              yc(i,j)=yc(i,j-1)+gridc
  8   continue
  7   continue

*     Calculate x.

      do 9 i=1,xg+1
      do 10 j=1,xg+1
        if (i .eq. 1) then
              xc(i,j)=xmin
        else if (i .ne. 1) then
              xc(i,j)=xc(i-1,j)+gridc
        end if
  10  continue
  9   continue

*     Calculate radius to each grid point.

      do 30 i=1,xg+1
      do 30 j=1,xg+1
              rcc(i,j)=sqrt((xc(i,j)**2)+(yc(i,j)**2))
  30  continue

*     Calculate tangential velocities.

      do 50 i=1,xg+1
      do 50 j=1,xg+1
      if (rcc(i,j) .ge. rc) then
              vtanc(i,j)=(vmax*rc)/rcc(i,j)
      else if (rcc(i,j) .lt. rc) then
              vtanc(i,j)=(vmax*rcc(i,j))/rc
      end if
  50  continue

*     Calculate theta.

      do 70 i=1,xg+1
      do 70 j=1,xg+1
      if (xc(i,j) .ne. 0 .and. yc(i,j) .ne. 0) then
              thetac(i,j)=(atan(yc(i,j)/xc(i,j))*(180/pi))
      else if (xc(i,j) .eq. 0 .and. yc(i,j) .eq. 0) then
              thetac(i,j)=0.0
      end if
  70  continue

*     Calculate gamma (angle of tangential winds).

      do 85 i=1,xg+1
```

```
        do 85 j=1,xg+1
        if (xc(i,j) .gt. 0 .and. yc(i,j) .gt. 0) then
                gammac(i,j)=90+thetac(i,j)
        else if (xc(i,j) .lt. 0 .and. yc(i,j) .gt .0) then
                gammac(i,j)=270+thetac(i,j)
        else if (xc(i,j) .lt. 0 .and. yc(i,j) . lt. 0) then
                gammac(i,j)=thetac(i,j)-90
        else if (xc(i,j) .gt. 0 .and. yc(i,j) .lt .0) then
                gammac(i,j)=90+thetac(i,j)
        else if (xc(i,j) .eq. 0 .and. yc(i,j) .gt. 0) then
                gammac(i,j)=180
        else if (xc(i,j) .eq. 0 .and. yc(i,j) .lt. 0) then
                gammac(i,j)=0
        else if (xc(i,j) .gt. 0 .and. yc(i,j) .eq. 0) then
                gammac(i,j)=90
        else if (xc(i,j) .lt. 0 .and. yc(i,j) .eq. 0) then
                gammac(i,j )=270
        end if
  85    continue

*       Calculate the vector components of the tangential wind.

        do 100 i=1,xg+1
        do 100 j=1,xg+1
                utc(i,j)=vtanc(i,j)*(cos(gammac(i,j)*pi/180))
                vtc(i,j)=vtanc(i,j )*(sin(gammac(i,j)*pi/180))
                utac(i,j)=utc(i,j)*.3
                vtac(i,j)=vtc(i,j)*.3
  100   continue

*       Calculate radial velocities.

        do 120 i=1,xg+1
        do 120 j=1,xg+1
        if (rcc(i,j) .ge. rc) then
                vradc(i,j)=(vrmax*rc)/rcc(i,j)
        else if (rcc(i,j) .lt. rc) then
                vradc(i,j)=(vrmax*rcc(i,j))/rc
        end if
  120   continue

*       Calculate delta (angle of inflow).

        do 121 i=1,xg+1
        do 121 j=1,xg+1
        if (xc(i,j) .gt. 0 .and. yc(i,j) .gt. 0) then
                deltac(i,j)=180+thetac(i,j)
        else if (xc(i,j) .lt. 0 .and. yc(i,j) .gt .0) then
                deltac(i,j)=thetac(i,j)
        else if (xc(i,j) .lt. 0 .and. yc(i,j) .lt.  0) then
                deltac(i,j)=thetac(i,j)
        else if (xc(i,j) .gt. 0 .and. yc(i,j) .lt .0) then
                deltac(i,j)=180+thetac(i,j)
        else if (xc(i,j) .eq. 0 .and. yc(i,j) .gt. 0) then
                deltac(i,j)=270
        else if (xc(i,j) .eq. 0 .and. yc(i,j) .lt. 0) then
                deltac(i,j)=90
```

```
        else if (xc(i,j) .gt. 0 .and. yc(i,j) .eq. 0) then
                deltac(i,j)=180
        else if (xc(i,j) .lt. 0 .and. yc(i,j) .eq. 0) then
                deltac(i,j)=0
        end if
  121   continue

*       Calculate the vector components of the radial wind.

        do 130 i=1,xg+1
        do 130 j=1,xg+1
                urc(i,j)=vradc(i,j)*(cos(deltac(i,j)*pi/180))
                vrc(i,j)=vradc(i,j)*(sin(deltac( i,j)*pi/180))
                urac(i,j)=urc(i,j)*.3
                vrac(i,j)=vrc(i,j)*.3
                utrc(i,j)=utac(i,j)+urac(i,j)
                vtrc(i,j)=vtac(i,j)+vrac(i,j)
  130   continue

*     Calculate vortex wind speed.

        do 135 i=1,xg+1
        do 135 j=1,xg+1
                utranc(i,j)=0
                vtranc(i,j)=vforw*(sin(5156.6202*pi/180))
                vtranc(i,j)=vtranc(i,j)*.3
                ucomc(i,j)=utranc(i,j)+utrc(i,j)
                vcomc(i,j)=vtranc(i,j)+vtrc(i,j)
                vtornc(i,j)=sqrt(((ucomc(i,j)/.3)**2)+
     $            ((vcomc(i,j)/.3)**2))
  135   continue

  138   yg=ymax*2/grid

*     Create color contour plot of vortex wind field.

      write(*,*)"Printing contour plot...(contour.ps)"
      call newdev('contour.ps',7)
      call psinit(.true.)
      call factor(0.6)
      call plot(4.,120.,-3)
      call border(300.,300.,0000,1111,4,2,3,4)
      call keknum(-1.,150.,5.,0.,90.,-1,1)
      call keknum(-1.,300.,5.,200.,90.,-1,1)
      call keknum(-1.,-8.,5.,-200.,0.,-1,2)
      call keknum(150.,-8.,5.,0.,0.,-1,1)
      call keknum(300.,-8.,5.,200.,0.,-1,1)
      call keksym(150.,-20.,7.,12hDistance (m),0.,12,1)
      call keksym(-8.,150.,7.,12hDistance (m),90.,12,1)
      call setlw(1.0)
      call arrow(150.,305.,150.,330.,5.,30.,0)
      call setlw(0.)
      if(answer.eq.'1')then
      call keksym(160.,320.,6.,7hTornado,0.,7,0)
      call keksym(160.,310.,6.,8hMovement,0.,8,0)
      else
      call keksym(160.,320.,6.,12hThunderstorm,0.,12,0)
```

```
      call keksym(160.,310.,6.,8hMovement,0 .,8,0)
      end if
      call factor(0.3)
      call setlw(2.0)
      call arrow(615.,190.,645.,190.,5.,30.,0)
      call setlw(0.)
      call keksym(627.,200.,10.,6hScale:,0.,6,1)
      call keknum(627.,170.,10.,60.,0.,-1,1)
      call keksym(627.,157.,10.,3hm/s,0.,3,1)
      call plot(0.,0.,-3)

      call factor(0.6)

      nval=10
      do 150 n=1,nval
            cval(n)=(n-1)*10.
      if(cval(n).le.10.)then         !grey
            color(1,n)=.8
            color(2,n)=.8
            color(3,n)=.8
      else if(cval(n).le.20.)then    !dark blue
            color(1,n)=0.
            color(2,n)=0.
            color(3,n)=1.
      else if(cval(n).le.30.)then    !medium blue
            color(1,n)=0.
            color(2,n)=.5
            color(3,n)=1.
      else if(cval(n).le.40.)then    !aqua
            color(1,n)=0.
            color(2,n)=1.
            color(3,n)=1.
      else if(cval(n).le.50.)then    !green
            color(1,n)=0.
            color(2,n)=1.
            color(3,n)=0.
      else if(cval(n).le.60.)then    !yellow
            color(1,n)=1.
            color(2,n)=1.
            color(3,n)=0.
      else if(cval(n).le.70.)then    !light orange
            color(1,n)=1.
            color(2,n)=.8
            color(3,n)=0.
      else if(cval(n).le.80.)then    !dark orange
            color(1,n)=1.
            color(2,n)=.5
            color(3,n)=0.
      else if(cval(n).gt.80.)then    !red
            color(1,n)=1.
            color(2,n)=0.
            color(3,n)=0.
      end if
      continue
  150 call concolr(vtornc,41,41,41,300.,300.,cval,
     $  color,nval,1,999.)
```

```
*     Plot vortex wind field on top of contour plot.

      call factor(0.45)
        do 155 i=1,yg+1,d
        do 155 j=1,yg+1,d
                xpp(i,j)=xss(i,j)+ucom(i,j)
                ypp(i,j)=yss(i,j)+vcom(i,j)
                call arrow(xss(i,j),yss(i,j),xpp(i,j),ypp(i,j),
     $            3.,30.,0)
  155   continue

*     Create legend.

      call rectfilc(10.,-40.,50.,-40.,10.,.8,.8,.8)
      call rectfilc(50.,-40.,90.,-40.,10.,0.,0.,1.)
      call rectfilc(90.,-40.,130.,-40.,10.,0.,.5,1.)
      call rectfilc(130.,-40.,170.,-40.,10.,0.,1.,1.)
      call rectfilc(170.,-40.,210.,-40.,10.,0.,1.,0.)
      call rectfilc(210.,-40.,250.,-40.,10.,1.,1.,0.)
      call rectfilc(250.,-40.,290.,-40.,10.,1.,.8,0.)
      call rectfilc(290.,-40.,330.,-40.,10.,1.,.5,0.)
      call rectfilc(330.,-40.,370.,-40.,10.,1.,0.,0.)
      call keknum(10.,-47.,5.,0.,0.,-1,0)
      call keknum(50.,-47.,5.,10.,0.,-1,0)
      call keknum(90.,-47.,5.,20.,0.,-1,0)
      call keknum(130.,-47.,5.,30.,0.,-1,0)
      call keknum(170.,-47.,5.,40.,0.,-1,0)
      call keknum(210.,-47.,5.,50.,0.,-1,0)
      call keknum(250.,-47.,5.,60.,0.,-1,0)
      call keknum(290.,-47.,5.,70.,0.,-1,0)
      call keknum(330.,-47.,5.,80.,0.,-1,0)
      call keksym(370.,-47.,5.,3hm/s,0.,3,0)

      if(answer.eq.'1')then
      call keksym(10.,-70.,7.,27hMax. tangential velocity = ,0.,
     $      27,0)
      call keknum(999.,999.,7.,vmax,0., -1,0)
      call keksym(999.,999.,7.,4h m/s,0.,4,0)
      call keksym(10.,-80.,7.,23hRadius of max. winds = ,0.,23,0)
      call keknum(999.,999.,7.,rc,0., -1,0)
      call keksym(999.,999.,7.,2h m,0.,2,0)
      call keksym(10.,-90.,7.,23hMax. radial velocity = ,0.,23,0)
      call keknum(999.,999.,7.,vrmax,0.,-1,0)
      call keksym(999.,999.,7.,4h m/s,0.,4,0)
      call keksym(10.,-100.,7.,16hForward speed = ,0.,16,0)
      call keknum(999.,999.,7.,vforw,0., -1,0)
      call keksym(999.,999.,7.,4h m/s,0.,4,0)
      call keksym(10.,-110.,7.,21hMaximum wind speed = ,0.,21,0)
      call keknum(999.,999.,7.,vtornmax,0., -1,0)
      call keksym(999.,999.,7.,7h m/s = ,0.,7,0)
      call keknum(999.,999.,7.,vtornmaxmph,0., -1,0)
      call keksym(999.,999.,7.,4h mph,0.,4,0)
      else
      call keksym(10.,-70.,7.,27hMax. microburst velocity = ,0.,
     $      27,0)
      micromax=-vrmax
      call keknum(999.,999.,7.,micromax,0., -1,0)
```

```
      call keksym(999.,999.,7.,4h m/s,0.,4,0)
      call keksym(10.,-80.,7.,23hWidth of damage path = ,0.,23,0)
      call keknum(999.,999.,7.,rc,0.,-1,0)
      call keksym(999.,999.,7.,2h m,0.,2,0)
      call keksym(10.,-90.,7.,16hForward speed = ,0.,16,0)
      call keknum(999.,999.,7.,vforw,0.,-1,0)
      call keksym(999.,999.,7.,4h m/s,0.,4,0)
      call keksym(10.,-100.,7.,21hMaximum wind speed = ,0.,21,0)
      call keknum(999.,999.,7.,vtornmax,0.,-1,0)
      call keksym(999.,999.,7.,7h m/s = ,0.,7,0)
      call keknum(999.,999.,7.,vtornmaxmph,0.,-1,0)
      call keksym(999.,999.,7.,4h mph,0.,4,0)
      end if
      if(answer.eq.'1')then
      if((vtornmaxmph.ge.40.).and.(vtornmaxmph.le.72.))then
      call keksym(10.,-120.,7.,10hF0 Tornado,0.,10,0)
      else if((vtornmaxmph.ge.73.).and.(vtornmaxmph.le.112.))then
      call keksym(10.,-120.,7.,10hF1 Tornado,0.,10,0)
      else if((vtornmaxmph.ge.113.).and.(vtornmaxmph.le.157.))then
      call keksym(10.,-120.,7.,10hF2 Tornado,0.,10,0)
      else if((vtornmaxmph.ge.158.).and.(vtornma xmph.le.206.))then
      call keksym(10.,-120.,7.,10hF3 Tornado,0.,10,0)
      else if((vtornmaxmph.ge.207.).and.(vtornmaxmph.le.260.))then
      call keksym(10.,-120.,7.,10hF4 Tornado,0.,10,0)
      else if((vtornmaxmph.ge.261.).and.(vtornmaxmph.le.318.))then
      call keksym(10.,-120.,7.,10hF5 Tornado,0.,10,0)
      else if(vtornmaxmph.ge.319)then
      call keksym(10.,-120.,7.,10hF6 Tornado,0.,10,0)
      end if
      else
      call keksym(10.,-120.,7.,10hMicroburst,0.,10,0)
      end if
      call factor(0.6)
      call keksym(275.,-125.,5.,8hTornTree,0.,8,0)

      call plotnd

      return
      end

*****CRITICALWIND********************************************************
*                                                                       *
*   Calculates the critical wind speed for stem breakage with distance *
*   into the stand (in tree heights) and writes to file 'critwind.dat'.*
*                                                                       *
*************************************************************************

      subroutine critwind(option,gapsize,grid,selection)

      character option              !choice between loblolly and Scots
                                          !pine
      character selection           !the menu selection
      real z(0:50)                  !height of each one -meter segment
                                          !(m)
      integer distedge(300,300)     !distance fro m forest edge in tree
                                          !heights
      real gustmax(300,300)         !used to calculate gust factor
```

```
real gustmeanedge(300,300)    !used to calculate gust factor
real gustfac(300,300)         !factor to include gustiness of wind
real gapmax(300,300)          !used to calculate gap factor
real gapmeanedge(300,300)     !used to calculate gap factor
real gapfac(300,300)          !factor to include size of the
                                    !upwind gap
real stemres(300,300)         !resistance to stem breakage (Nm)
real area(300,300,0:50)       !area of each one-meter segment
                                    !m^2
real ms(300,300,0:50)         !mass of each one-meter segment (kg)
real fwind(300,300,0:50)      !force due to the wind (N)
real fwindtot(300,300)        !sum of fwind over the tree height
                                    !(N)
real fmass(300,300,0:50)      !force due to gravity (N)
real ltop(0:50)               !distance from tree top (m)
real horizd(300,300,0:50)     !horizontal displacement of the stem
                                    !(m)
real h(300,300)               !height of the tree (m)
real cd                       !drag coefficient
real rho                      !density of air (kg/m^3)
real g                        !gravitational constant (m/s^2)
real dbh(300,300)             !diameter at breast height (m)
real a(300,300)               !distance from ground to crown
                                    !center (m)
real b(300,300)               !distance from tree top to crown
                                    !center (m)
real crhgt(300,300)           !height from ground to crown base
                                    !(m)
real stemhgt(300,300)         !total height - crhgt, stem height
                                    !(m)
real moe                      !modulus of elasticity (Pa)
real mor                      !modulus of rupture (Pa)
real ami(300,300)             ! area moment of inertia (m^4)
real gh(300,300)              !spacing between trees in tree
                                    !heights
real grid                     !tree spacing (m)
real gapsize                  !size of the upwind gap in tree
                                    !heights
real ucrarea(300,300,0:50)    !unstreamlined crown area (m^2)
real starea(300,300,0:50)     !stem area (m^2)
real crarea(300,300,0:50)     !streamlined crown area (m^2)
real st(300,300,0:50)         !streamlining function
real v                        !the wind speed (m/s)
real u(0:50)                  !the vertical wind profile (m/s)
real zn(300,300)              !roughness length
real ht,db                    !height and dbh
real taper                    !height (m)/dbh (m)
real tmax(300,300)            !maximum turning moment (Nm)
real tmom(300,300,0:50)       !turning moment for segment (Nm)


write(*,*)"Calculating the critical wind speeds"
write(*,*)"...(critwind.dat)"
write(*,*)"Printing plot of fallen trees...(trees.ps)"

pi=3.14
rho=1.226
```

```
      g=9.8

      do 400 m=1,10
      do 400 n=1
                distedge(m,n)=10-m
400   continue

      if(option.eq.'1')then
            cd=0.2
            moe=2.41E9
            mor=49000000.
        do 410 m=1,10
        do 410 n=1
                open(1,file='lpage.dat')
                read(1,*)dbh(m,n),h(m,n)
                close(1)
          gh(m,n)=grid/h(m,n)
          ami(m,n)=pi*((dbh(m,n)**4)/64)
          crhgt(m,n)=(h(m,n)*0.40)
          stemhgt(m,n)=h(m,n)-crhgt(m,n)
          a(m,n)=(crhgt(m,n)/2)+stemhgt(m,n)
          b(m,n)=h(m,n)-a(m,n)
410       continue

        do 415 m=1,10
        do 415 n=1
        do 420 k=0,h(m,n)-1
                open(4,file='lpucrarea.dat')
                read(4,*)ucrarea(m,n,k)
                open(5,file='lpstarea.dat')
                read(5,*)starea(m,n,k)
                open(6,file='lpmass.dat')
                read(6,*)ms(m,n,k)
420       continue
          close(4)
          close(5)
          close(6)
415       continue
      end if

      if(option.eq.'2')then
            cd=0.29
            moe=7E9
            mor=39100000.
          do 425 m=1,10
          do 425 n=1
            if(selection.eq.'1')then
                open(1,file='spage35.dat')
                read(1,*)dbh(m,n),h(m,n)
                close(1)
            else if(selection.eq.'2')then
                open(2,file='spage65.dat')
                read(2,*)dbh(m,n),h(m,n)
                close(2)
            else if(selection.eq.'3')then
                open(3,file='spage95.dat')
                read(3,*)dbh(m,n),h(m,n)
```

```
             close(3)
         else if(selection.eq.'4')then
             open(4,file='spage120.dat')
             read(4,*)dbh(m,n),h(m,n)
             close(4)
         end if
       gh(m,n)=grid/h(m,n)
       ami(m,n)=pi*((dbh(m,n)**4)/64)
       crhgt(m,n)=(h(m,n)*0.42)
       stemhgt(m,n)=h(m,n)-crhgt(m,n)
       a(m,n)=(crhgt(m,n)/2)+stemhgt(m,n)
       b(m,n)=h(m,n)-a(m,n)
425    continue

     do 430 m=1,10
     do 430 n=1
     do 435 k=0,h(m,n)-1
       if(selection.eq.'1')then
             open(5,file='spucrarea35.dat')
             read(5,*)ucrarea(m,n,k)
             open(6,file='spstarea35.dat')
             read(6,*)starea(m,n,k)
             open(7,file='spmass35.dat')
             read(7,*)ms(m,n,k)
         else if(selection.eq.'2')then
             open(8,file='spucrarea65.dat')
             read(8,*)ucrarea(m,n,k)
             open(9,file='spstarea65.dat')
             read(9,*)starea(m,n,k)
             open(10,file='spmass65.dat')
             read(10,*)ms(m,n,k)
         else if(selection.eq.'3')then
             open(11,file='spucrarea95.dat')
             read(11,*)ucrarea(m,n,k)
             open(12,file='spstarea95.dat')
             read(12,*)starea(m,n,k)
             open(13,file='spmass95.dat')
             read(13,*)ms(m,n,k)
         else if(selection.eq.'4')then
             open(14,file='spucrarea120.dat')
             read(14,*)ucrarea(m,n,k)
             open(15,file='spstarea120.dat')
             read(15,*)starea(m,n,k)
             open(16,file='spmass120.dat')
             read(16,*)ms(m,n,k)
         end if
435    continue
         close(5)
         close(6)
         close(7)
         close(8)
         close(9)
         close(10)
         close(11)
         close(12)
         close(13)
         close(14)
```

```
            close(15)
            close(16)
430     continue
      end if

        do 480 m=1,10
        do 480 n=1
        gustmeanedge(m,n)=(((1.7239*gh(m,n)+0.0316)**(0.))
     $  *(-0.68*gh(m,n)+0.4785))+(0.68*gh(m,n)-0.0385)
        gustmax(m,n)=(((1.1127*gh(m,n)+0.0311)**(distedge(m,n)))
     $  *(-1.273*gh(m,n)+0.9701))+(2.7193*gh(m,n)-0.061)
        gustfac(m,n)=gustmax(m,n)/gustmeanedge(m,n)
        gapmeanedge(m,n)=(0.001+0.001*(10.**0.562))/0.00465
        gapmax(m,n)=(0.0072+0.0064*(gapsize**0.3467))/0.0214
        gapfac(m,n)=gapmax(m,n)/gapmeanedge(m,n)
        stemres(m,n)=(pi/32)*mor*(dbh(m,n)**3)
480     continue

      open(10,file='critwind.dat')
      if(option.eq.'1')then
            write(10,*)"Loblolly pine"
      else
            write(10,*)"Scots pine"
      end if

      ht=-1.e20
      db=-1.e20
      do 485 m=1,10
      do 485 n=1
            ht=amax1(ht,h(m,n))
              db=amax1(db,dbh(m,n))
485   continue

      taper=ht/db
      db=db*100

      write(10,700)ht
700   format(' height= ',f4.1,'m')
      write(10,701)db
701   format(' diameter at breast height= ',f4.1,'cm')
      write(10,702)grid
702   format(' tree spacing= ',f4.1,'m')
      write(10,703)taper
703   format(' tree taper= ',f5.1)
      write(10,704)gapsize
704   format(' upwind gapsize= ',f4.1,' tree heights')

      write(10,*)

        do 495 m=10,1,-1
        do 495 n=1
        v=1.
500       tmax(m,n)=0.0
          fwindtot(m,n)=0.0
          zn(m,n)=0.06*h(m,n)

*     Calculate the total force due to the wind.
```

```
      do 497 o=0,h(m,n)-1
        z(o)=0.5+o

*     Calculate wind profile (u).

        u(o)=v*(log(z(o)/(zn(m,n)))/
     $        log(h(m,n)/(zn(m,n))))
        if(u(o).lt.0.)then
          u(o)=0.
        end if

*     Streamline the crown area and add it to the stem are a to obtain
*     total area distribution.

          if(u(o).le.11.0)then
            crarea(m,n,o)=ucrarea(m,n,o)*0.80
            area(m,n,o)=crarea(m,n,o)+starea(m,n,o)
        else if(u(o).ge.20.0)then
            crarea(m,n,o)=ucrarea(m,n,o)*0.40
            area(m,n,o)=crarea(m,n,o)+starea(m,n,o)
        else if(u(o).gt.11.0 .and. u(o).lt.20.0)then
            st(m,n,o)=(10/u(o))-0.1
            crarea(m,n,o)=ucrarea(m,n,o)*st(m,n,o)
            area(m,n,o)=crarea(m,n,o)+starea(m,n,o)
        end if
          fwind(m,n,o)=(0.5*cd*rho*area(m,n,o)*(u(o)**2))
          fwindtot(m,n)=fwindtot(m,n)+fwind(m,n,o)
  497   continue

        do 498 k=0,h(m,n)-1
        z(k)=0.5+k

*     Calculate wind profile (vtree).

        u(k)=v*(log(z(k)/(zn(m,n)))/
     $        log(h(m,n)/(zn(m,n))))
        if(u(k).lt.0.)then
          u(k)=0.
        end if

          ltop(k)=h(m,n)-z(k)


*     Streamline the crown area and add it to the stem area to obtain
*     total area distribution.

         if(u(k).le.11.0)then
            crarea(m,n,k)=ucrarea(m,n,k)*0.80
            area(m,n,k)=crarea(m,n,k)+starea(m,n,k)
        else if(u(k).ge.20.0)then
            crarea(m,n,k)=ucrarea(m,n,k)*0.40
            area(m,n,k)=crarea(m,n,k)+starea(m,n,k)
        else if(u(k).gt.11.0 .and. u(k).lt.20.0)then
            st(m,n,k)=(10/u(k))-0.1
            crarea(m,n,k)=ucrarea(m,n,k)*st(m,n,k)
            area(m,n,k)=crarea(m,n,k)+starea(m,n,k)
```

```
        end if
        fwind(m,n,k)=(0.5*cd*rho*area(m,n,k)*(u(k)**2))
        fmass(m,n,k)=g*ms(m,n,k)
        if(z(k) .ge. a(m,n))then
        horizd(m,n,k)=(fwindtot(m,n)*(a(m,n)**2)*h(m,n)*(3 -(a(m,n)
     $  /h(m,n))-(3*ltop(k)/h(m,n)))))/(6*moe*ami(m,n))
        else if(z(k) .lt. a(m,n))then
        horizd(m,n,k)=(fwindtot(m,n)*(a(m,n)**3)*(2 -3*((ltop(k)-b(m,n))
     $  /a(m,n))+(((ltop(k)-b(m,n))**3)/a(m,n)**3)))/(6*moe*ami(m,n))
        end if
        tmom(m,j,k)=gustfac(m,j)*gapfac(m,j)*(fwind(m,j,k)*z(k)
     $          +fmass(m,j,k)*horizd(m,j,k))
        tmax(m,n)=tmax(m,n)+tmom(m,n,k)
        tmax(m,n)=tmax(m,n)-tmom(m,n,0)
  498   continue

        if (tmax(m,n) .ge. stemres(m,n)) then
*           write(10,*)"distance from edge=",distedge(m,n),
*     $          "     critical wind speed=",v
            write(10,800)distedge(m,n),v
      else
            v=v+0.1
            goto 500
        end if
  495   continue

  800 format('distance from edge= ',i1,'h',
     $           '     critical wind speed= ',f6.1,' m/s')
      close(10)

      write(*,*)"Printing plot of fallen trees...(trees.ps)"
      return

      end

***LOBLOLLYDATA*********************************************************     *
*
*   Takes user's input of height, dbh and spacing and creates the age, *
*   area and mass distribution files for use in subroutine LOBLOLLY.   *
************************************************************************

      subroutine loblollydata(spacing)

      real r,c,e,p,b,a        !coefficients based on tree species
      real htft               !tree height in feet
      real dbhcm              !dbh in cm
      real d                  !dbh in in.
      real h(0:100)           !height in in. of one-meter segments
      real dbhm               !dbh in meters
      real htm                !tree height in meters
      real dcm(0:100)         !diameter in cm at each one-meter height
      real dm(0:100)          !same as above in meters
      real din(0:100)         !same as above in inches
      real f                  !diameter at 17.3 feet
      real is,ib,it,im        !indicator variables
      real spacing            !tree spacing in meters
      real rhttri(100)        !height of triangle in crown (m)
```

```
      real lbase(0:100)       !base of triangles in live crown (m)
      real dbase(0:100)       !base of triangles in dead crown (m)
      real base(0:100)        !base of triangles in crown (m)
      real ucrarea(100)       !unstreamlined crown area distribution
                                    !(m^2)
      real totucrarea         !total unstreamlined crown area (m^2)
      real af,bf              !coefficients for finding f for loblolly
      real starea(100)        !stem area distribution (m^2)
      real mass(100)          !stem and crown mass distribution (kg)
      real totstmasslb        !total stem mass in lbs.
      real totstmass          !total stem mass in kg
      real totstarea          !total stem area in m^2
      real stmass(100)        !stem mass distribution (kg)
      real crwoodmasslb,crwoodmass  !crown wood mass in lbs. and kg
      real crbarkmasslb,crbarkmass  !crown bark mass in lbs. and kg
      real crfolmasslb,crfolmass    !crown foliage mass in lbs. and kg
      real totcrmass          !total crown mass in kg
      real crmass(100)        !crown mass distribution in kg
      real wooda,woodb,woodc,barka  !coefficients for finding crown mass
      real barkb,barkc,fola,folb,folc !same as above
      integer livecrwnht      !live crown height (m)
      integer deadcrwnht      !dead crown height (m)
      integer crwnht          !total crown height (m)
      integer htbscrwn        !height of the crown base (m)
      integer ht              !tree height in meters
      integer htbslvcrwn      !height of the live crown base (m)
      integer httri(0:100)    !height of triangle in crown (m)


      r=25.29597
      c=0.5837
      e=221.45
      p=8.88273
      b=2.39522
      a=0.70372
      af=0.92022
      bf=-1.04015

*     Get user's inputs.

      write(*,*)"Enter height of the tree in meters."
      read(*,*)htm
      write(*,*)"Enter the diameter at breast height (cm)."
      read(*,*)dbhcm
      write(*,*)"Enter the spacing between the trees (m)."
      write(*,*)"*spacing/height must be between 0.75 and 0.45*"
      read(*,*)spacing

*     Convert height into feet and dbh into inches.

      ht=int(htm)
      htft=ht*3.28
      d=dbhcm*0.394
      dbhm=dbhcm/100

*     Calculate the diameter at 17.3 feet.
```

```
      f=d*(af+bf*(17.3/htft)**2)

*     Determine the indicator variables.

      do 10 i=0,ht
            h(i)=i*3.28

      if(h(i).lt.4.5)then
            is=1
            ib=0
            it=0
      else if(h(i).ge.4.5 .and. h(i).le.17.3)then
            is=0
            ib=1
            it=0
      else if(h(i).gt.17.3)then
            is=0
            ib=0
            it=1
      end if

      if(h(i).lt.(17.3+a*(htft-17.3)))then
            im=1
      else
            im=0
      end if

*     Calculate the stem diameter at the stem base and at every meter
*     above the base.

      din(i)=(is*((d**2)*(1+(c+e/d**3)*(((1-h(i)/htft)**r)-(1-4.5/
     $      htft)**r)/(1-(1-4.5/htft)**r)))+ib*((d**2)-((d**2)-(f**2))*
     $  (((1-4.5/htft)**p)-(1-h(i)/htft)**p)/(((1-4.5/htft)**p)-
     $  (1-17.3/htft)**p))+it*((f**2)*(b*(((h(i)-17.3)/(htft-17.3))-1)
     $  **2+(im*((1-b)/a**2)*(a-((h(i)-17.3)/(htft-17.3)))**2))))**0.5

      dcm(i)=din(i)/0.394
      dm(i)=dcm(i)/100
   10 continue

*     Calculate the stem area distribution and the total stem area.

      totstarea=0.0
      do 20 i=1,ht
            starea(i)=.5*(dm(i)+dm(i-1))
            totstarea=totstarea+starea(i)
   20 continue

*     Calculate the total stem mass.

      totstmasslb=0.17567*((d**2)**1.00751)*htft**0.95925
      totstmass=totstmasslb*.4536

*     Calculate the stem mass distribution.

      do 30 i=1,ht
            stmass(i)=(starea(i)/totstarea)*totstmass
```

```
   30  continue

*      Determine the live and dead crown heights.

       livecrwnht=ht*0.30
       crwnht=ht*0.40
       deadcrwnht=crwnht-livecrwnht
       htbscrwn=ht-crwnht
       htbslvcrwn=ht-livecrwnht

*      Calculate the bases of the triangles in the crown.


       do 40 i=0,ht
         if(i.lt.htbslvcrwn)then
             httri(i)=0.0
             lbase(i)=0.0
         else if(i.eq.htbslvcrwn)then
             httri(i)=livecrwnht
             lbase(i)=spacing
         else
             httri(i)=ht-i
             rlivecrwnht=real(livecrwnht)
             rhttri(i)=real(httri(i))
             lbase(i)=(rhttri(i)/rlivecrwnht)*spacing
         end if
   40  continue

       do 50 i=ht,0,-1
         if(i.gt.htbslvcrwn.or.i.lt.htbscrwn)then
             httri(i)=0
             dbase(i)=0.0
         else if(i.eq.htbslvcrwn)then
             httri(i)=deadcrwnht
             dbase(i)=0.0
         else
             httri(i)=httri(i+1)-1
             rdeadcrwnht=real(deadcrwnht)
             rhttri(i)=real(httri(i))
             dbase(i)=(rhttri(i)/rdeadcrwnht)*spacing
         end if
   50  continue

       do 60 i=0,ht
             base(i)=lbase(i)+dbase(i)
   60  continue

*      Calculate unstreamlined crown area distribution and total
*      unstreamlined crown area.

       totucrarea=0.0
       do 70 i=1,ht
             ucrarea(i)=0.5*(base(i)+base(i-1))
             totucrarea=totucrarea+ucrarea(i)
   70  continue

       wooda=1.735217
```

```
      woodb=3.492293
      woodc=-1.243386
      barka=1.203148
      barkb=3.023912
      barkc=-1.136030
      fola=3.652443
      folb=2.864732
      folc=-1.454774

*     Calculate the masses of the crown components and the total crown *
      mass.

      crwoodmasslb=exp(wooda+woodb*log(d)+(woodc)*log(htft))
      crwoodmass=crwoodmasslb*.4536
      crbarkmasslb=exp(barka+barkb*log(d)+(barkc)*log(htft))
      crbarkmass=crbarkmasslb*.4536
      crfolmasslb=exp(fola+folb*log(d)+(folc)*log(htft))
      crfolmass=crfolmasslb*.4536
      totcrmass=crwoodmass+crbarkmass+crfolmass

*     Calculate the crown mass distribution.

      do 80 i=1,ht
            crmass(i)=(ucrarea(i)/totucrarea)*totcrmass
  80  continue

*     Calculate the total tree mass.

      do 90 i=1,ht
            mass(i)=crmass(i)+stmass(i)
  90  continue

*     Write data to files to be used by subroutine LOBLOLLY.

      open(10,file='lpstarea.dat')
      open(20,file='lpucrarea.dat')
      open(30,file='lpmass.dat')
      open(40,file='lpage.dat')
      write(40,*)dbhm,',',htm

      do 100 i=1,ht
            write(10,*)starea(i)
            write(20,*)ucrarea(i)
            write(30,*)mass(i)
  100 continue

      close(10)
      close(20)
      close(30)
      close(40)

      return
      end
```

```
*********************************************************
*                                                       *
*     PSPLOT - A PostScript Plotting Library            *
*     written by:  Kevin E. Kohler                      *
*     kevin@ocean.nova.edu                              *
*     http://www.nova.edu/ocean/psplot.html             *
*                                                       *
*********************************************************
```