# PHOENIX RapidFire 4.0 Convection and Ember Dispersal Model

## Technical Report

A report prepared by:
Derek Chong (University of Melbourne/Bushfire CRC)
Kevin Tolhurst (University of Melbourne)
Thomas Duff (Bushfire CRC)

| | |
|---|---|
| Theme: | Understanding Risk |
| Project: | Fire Impact and Risk Evaluation Decision Support Tool |
| Component: | Enhancement of Fire Behaviour Models |
| Milestone: | 3.2.3 |
| Date: | 16 December 2012 |

**Cover photo:** Spotfires ahead of the Cann River fire 1982. (Source: Ross Runnalls, DSE)

This report was produced with financial support from the Bushfire CRC. This is not a published report and has had internal review, but not independent external peer review. Any opinions expressed in this report are those of the authors and not the University of Melbourne or the Bushfire CRC. Any use of original concepts, ideas or results in this report should be done in consultation with the authors.

## Summary

PHOENIX RapidFire is unique in the way it models and incorporates the spotting process in bushfires. There are three main components to the spotting process modelled: lofting, transport, and spotfire ignition. The method of modelling these three processes is described. Lofting is related to the convective strength of the fire and the amount of available ember material, transport is related to wind speed and direction, and spotfire ignition is related to the available fine fuel on the ground and its moisture content.

An absence of reliable ember transport and spotfire data has necessitated an empirical approach using the Black Saturday fires in Victoria for model development. As a result, there may be some aspects of the spotting process not well captured in PHOENIX, but to date, the results on new fires have been encouraging.

The effective rate of spread of bushfires can be a factor of two or three times greater if the spotting process is well developed compared with a fire, burning under generally similar conditions, but without spotting contributing to the effective rate of spread.

Being able to effectively model the spotting process has dramatically increased our ability to model bushfires in eucalypt forests. Modelling the conditions associated with the spotting process has also improved our ability to model potential house loss.

# Contents

## Introduction

The PHOENIX model incorporates elements of convection in its fire spread mechanism (Chong et al 2012). It identifies the likely location and power of convection columns that could establish over active parts of a fire. The convection model is based on the assumption that convection columns will form over the hottest areas of a fire. Large fires can have multiple convection columns at any one time. It uses this information, in conjunction with wind speed and direction to determine potential ember impact patterns resulting from these columns. Ember impacts are used to start spotfires as well as in asset loss calculations.

## Heat Segments

The first step in identifying potential convection columns or plumes is to locate relatively hot parts of a fires perimeter (heat segments). This has been defined as the perimeter segments having an average intensity value within in the top 25% of perimeter intensity values.

A PHOENIX fire perimeter consists of a series of ordered vertices, forming a polygon. A vertex intensity value for the identification of a heat segment is its average value for the preceding time step. Input and output data for PHOENIX are stored in a regular square grid, the size of which is specified by the user. Typically, these grid cells are about 100 to 200 m square. A vertex may traverse multiple grid cells during this time, each resulting in a different rate of spread and resulting intensity value.

A heat segment's intensity value is smoothed out using a running average over 10 perimeter points or 10% of the total perimeter point count, whichever is smaller. The start of a heat segment is flagged when its intensity exceeds the 25% threshold and ended when it drops below.

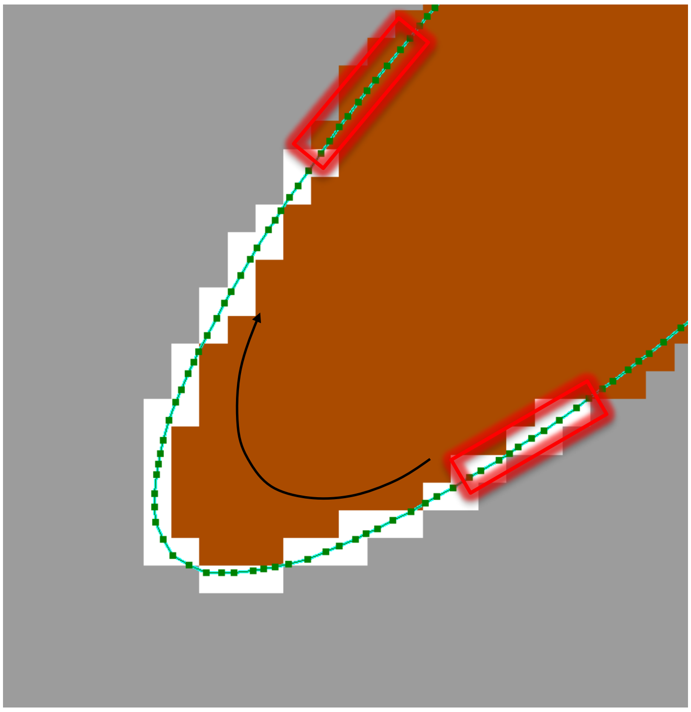

**Figure 1.** Fire perimeter is shown in green with dots representing the vertices. White pixels indicate a section that has been identified as being in the top 25% of perimeter intensities. The running 10 point average shown in red has traversed the perimeter, entering and exiting the 25% threshold value.

Multiple ignitions, topography, varying fuel types and load and variable weather can generate complex fire perimeter shapes.  Associated with complex fire shapes are often more than one active convection column as shown in Fig.2.

**Figure 2. Early stages of the Bunyip Ridge fire on 7 Feb 2009 showing 2 distinct convective centres.  Photo by Lex Wade**

The PHOENIX convection model allows for multiple distinct heat segments to be identified within a fire complex to reflect the multiple active fire fronts that can exist within a bushfire (e.g. Fig.3).

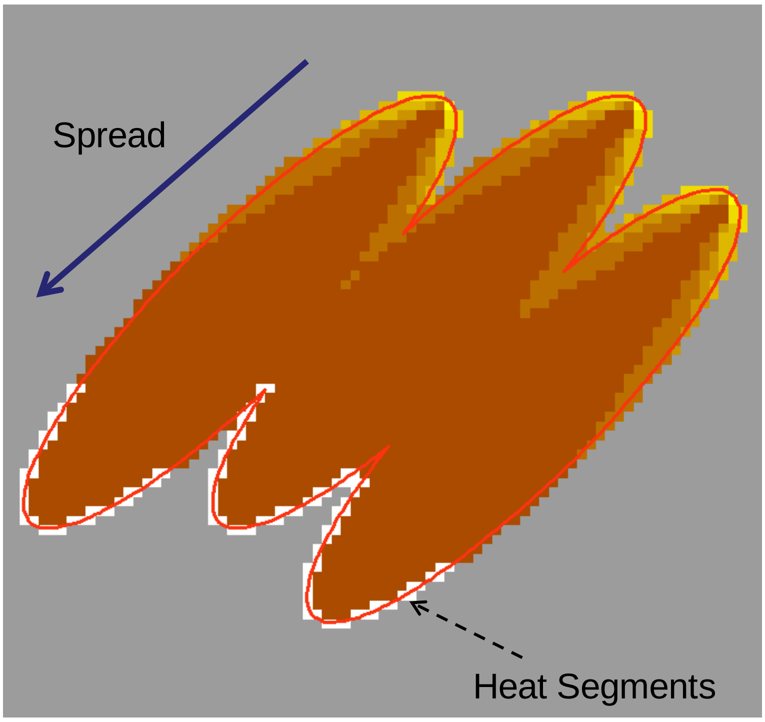

**Figure 3.** **Results from 3 separate ignitions that have merged into a complex fire shape with multiple active fronts, behaviour which regularly occurs in large bushfires where spotting, highly variable topography or fuel is prevalent.**

## Column Identification and Merging

Heat segments individually do not necessarily reflect the size and strength of a fire's dominant convection columns. For a single regularly shaped elliptical fire this is expected to be the case, however regularly shaped fire perimeters are the exception rather that the rule in naturally occurring bushfires. A large fast moving bushfire can have multiple active fronts and spotfires, each with their own associated convection columns (Figure 4). In these cases any convection columns formed are likely to be an aggregate of multiple local heat sources merged to produce locally dominant columns. The PHOENIX convection model attempts to reproduce this phenomenon by aggregating heat segments where they are deemed to be close enough to interact.

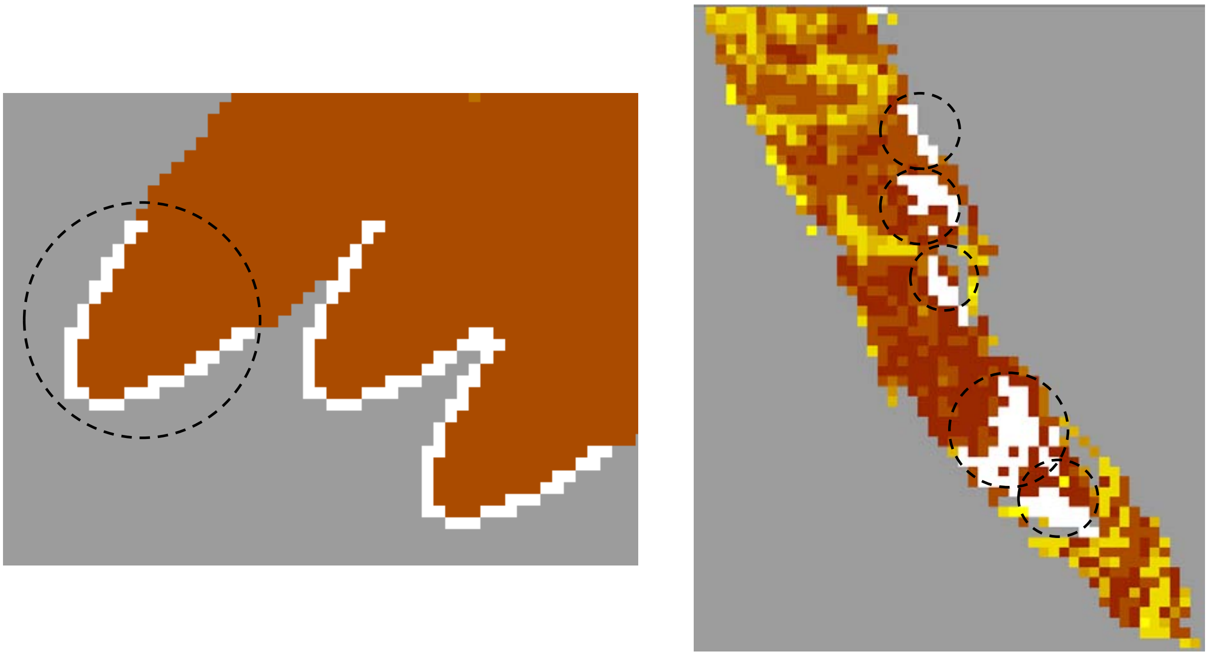

Figure 4. On the left, heat segments identified after 3 regularly shaped elliptical fires have merged compared to heat segments identified in a fire modelled in a natural landscape which has resulted in significant spotfires. Note the irregular shape and spatial distribution of the heat segments.

PHOENIX utilises a recursive merging algorithm that looks to aggregate heat segments where they are deemed to interact. Each heat segment is first identified as a potential convection column whose heat output is that of the perimeter vertices it contains. Vertex point intensity (kW/m) is converted to total heat output in kW by averaging the values between it and the next vertex and multiplying by their distance a part. Column convective output is expressed in megawatts (MW).

A column's centre of gravity is calculated using vertex locations weighted by their intensity. A minimum bounding box is calculated for the heat segment. The single segment column’s area of influence is expressed as its effective radius ($Column_{er}$) which is set as a function of its minimum bounding box dimensions($E$). This radius is inclusive of the area of convective indraught outside of the burning area which is assumed to be an additional 10% of the burning area radius.

$$Column_{er} = 1.1 \times \frac{E_h + E_w}{2}$$

Single heat segment columns are now sorted in descending order based on convective output. Starting with the strongest, each is tested against lesser columns and merged if within their effective radius.

When a heat segment is merged into a column, a new minimum bounding box is recalculated based on the vertices of the new heat segment, the additional heat output is added and the column's location or centroid is adjusted to reflect the new heat source.

The merging process is recursive and continues until all merging is complete, resorting after each pass.

**Figure 5.** A series of simulated images showing the progressive development of 3 fires in a flat, homogenous landscape (left to right). Circles show estimated position and strength (darker circles indicate greater convective output) of convection columns as they merge. Columns are initially independent, but as fires get progressively closer they merge to form a single column.

All heat segments, including those from separate fires (spotfires or multiple ignitions) are processed in the same recursive algorithm to capture the heat distribution in the landscape rather than a particular fire.

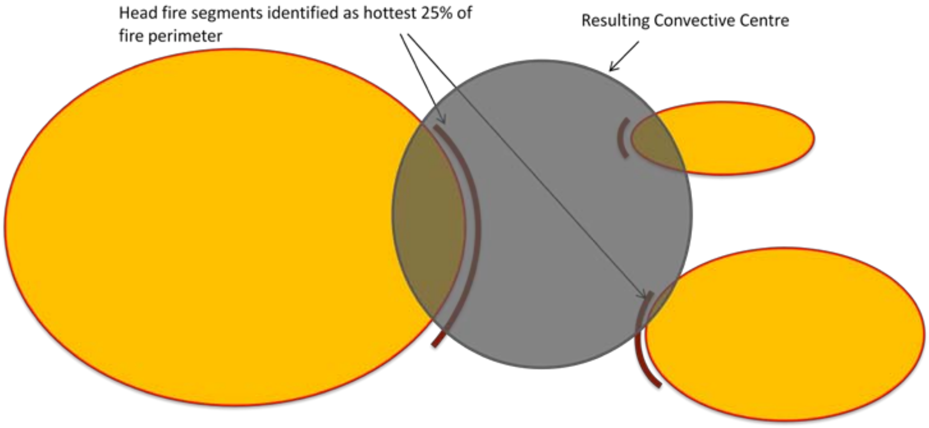

**Figure 6.** Diagram showing how the recursive merging algorithm processes heat segments from separate fires.

## Spotting

To capture the effect of spotting on fire spread, a deterministic spotfire model has been included in the PHOENIX model. The spotfire process is partitioned into three stages: ember launch, dispersal and ignition.

### Ember Launch

Ember launch is handled at the grid cell level rather than using the fires perimeter. Perimeter time steps and spatial resolution are variable whereas grid cells are remain constant during a simulation and a providing a consistent platform for spotting. When a point along a fire's perimeter impacts a cell, an ember launch event is triggered. Only cells that result in intensity values greater than the self extinction intensity (120 kW/m) are processed.

The number of embers available from a cell is scaled between the arbitrary range of 0 and 60 embers/m$^2$ based on the cell's bark load (McCarthy et al. 1999)

$$Available\ Embers = \frac{1}{1 + 108 * e^{(-1.2\ \times Bark\ Load)}}$$

This scaling mechanism assumes a greater amount of ember viable material with an increasing bark load.

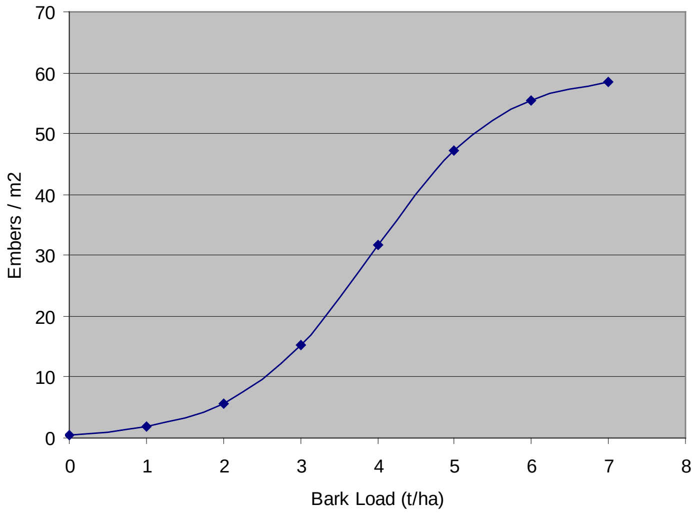

Figure 7. Graph showing the relationship between potential embers density and bark load.

Ember launches are only carried out on cells under the influence of a convective centre. It is assumed that the proportion of available embers launched is dependent on the convective strength of the centre affecting a cell.

$$Ember\ Porportion\ Launched = 1.032 - e^{-.000045\ \times \text{Convective Strength}}$$

A theoretical maximum ember 'Hang Time' in minutes is calculated based on the influencing column's convective strength. This value is intended to represent the maximum time a viable ember can remain aloft from a launch event. In its current implementation it is also used as a scaling mechanism that encapsulates an increased wind speed with altitude as experienced in the vicinity of the major fires of the 7th February 2009 in Victoria. The increased wind speed relative to surface was observed to a height of approximately 5,000 m.

$$Hang\ Time = 0.6\ \times Convective\ Strength \div 10000$$

Hang time values increase linearly with convective strength with maximum modelled values achieved in the Kilmore and Murrindindi fires being 28 and 36 minutes respectively.

## Ember Dispersal

The ember dispersal process can be conceptualised as a cloud of all the available embers from a cell launching simultaneously and being distributed by the prevailing winds. The embers are transported vertically by the convection column then horizontally by the prevailing winds.

Of all the embers launched, it is assumed that only a small proportion will reach the ground in a state that could result in a spotfire with the majority burning out before impacting the ground. It is assumed that the total number of viable embers reaching the ground from a launch is inversely proportional to the convective strength of the launching column. An increased hang time will result in more embers burning out before impact.

$$Total\ Viable\ Embers = Embers\ Launched\ \ \times e^{-9(\frac{Hang\ Time}{35})}$$

The transport of viable embers is modelled using the reference weather stream rather than the local terrain effected wind as it is assumed to better match the winds aloft.

Without knowing the ‘actual’ lofting heights, descent rates or vertical wind profile (which is assumed to be different from the sounding location and change during the day), it is not possible to capture the ‘real’ transport winds experienced by the spotting material. Instead, an empirically fitted ‘resultant’ spatial ember density distribution is calculated for each launch event and distributed across the landscape. A Weibull/bimodal distribution provided the best fit to observed spotting patterns (Sardoy, Consalvi et al. 2008). The bimodal distribution captured the medium to long distance spotting phenomenon better than traditional exponential decay models which only addresses short distance spotting patterns.

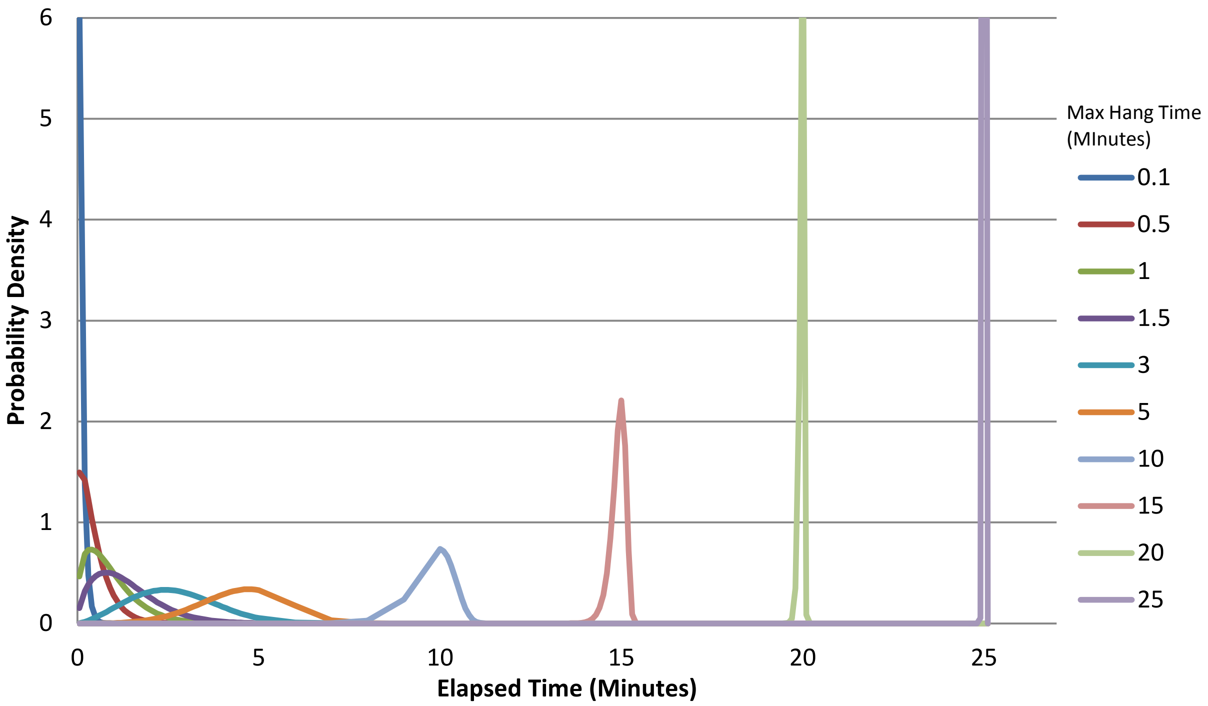

**Figure 8.** Bimodal ember impact pattern used in PHOENIX. Note, the cumulative probability version of the Weibull function is used to represent this pattern in Phoenix. This graph has been generated using the Weibull probability density function using the same shape and scale parameters as the cumulative function in order to show the change in ember impact pattern with an increasing hang time.

The Weibull cumulative distribution function (Figure 9) is used to describe the proportion of the available embers impacting along the principal axis of the ember impact pattern. This number includes the lateral ember impacts in the first case. For small convective values the majority of embers impacting are assumed to fall within a short time of launch, however as the hang time increases the majority of the smaller ember material is assumed to have burn out leaving only the larger or slower burning embers. The cumulative Weibull function used takes the following form:

$$Cumulative\ Weibull = 1 - e^{-\left(\frac{Minutes\ Elapsed}{Max\ Hang\ Time}\right)^{e(0.3\times Max\ Hang\ Time)}}$$

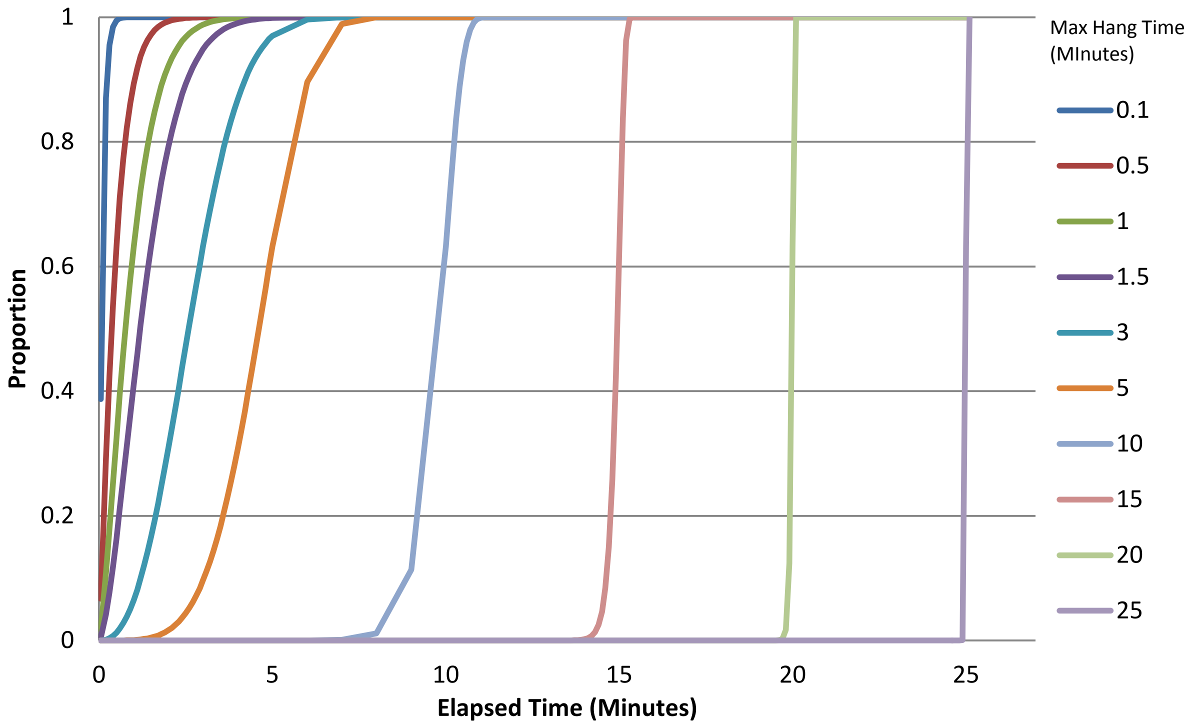

**Figure 9.** **Cumulative Weibull distribution function generates the bimodal ember impact pattern observed as fires increased in convective output on the 7th February 2009 and as described in (Sardoy, Consalvi et al. 2008)**

A Weibull function is also used to represent lateral ember distribution with ember hang time. Reconstructions of Black Saturday fires show a general transition from a widening spotfire impact pattern with distance, which subsequently narrows for longer distance impacts. It is assumed that these longer distance ignitions are caused by heavier slower burning spotting material which is less susceptible to turbulent flows in the plume which would widely distribute the smaller and lighter material.

Weibull function parameters were selected to produce an increasing lateral ember distribution for impacts up to 7 minutes, which then narrows to the 15 minute mark where it asymptotes to in order to capture discrete viable long distance ember impacts. The lateral spread standard deviation (m) is for 1 standard deviation around the point of impact along the principal axis of the ember impact pattern (Figure 11), mean = 0.

$$Lateral\ Spread_{std\ dev} = 1500 \times 0.275\ \times (Hang\ Time \div 8)^{1.2} \times e^{-(Hang\ Time \div 8)^{\ 2.2}}$$

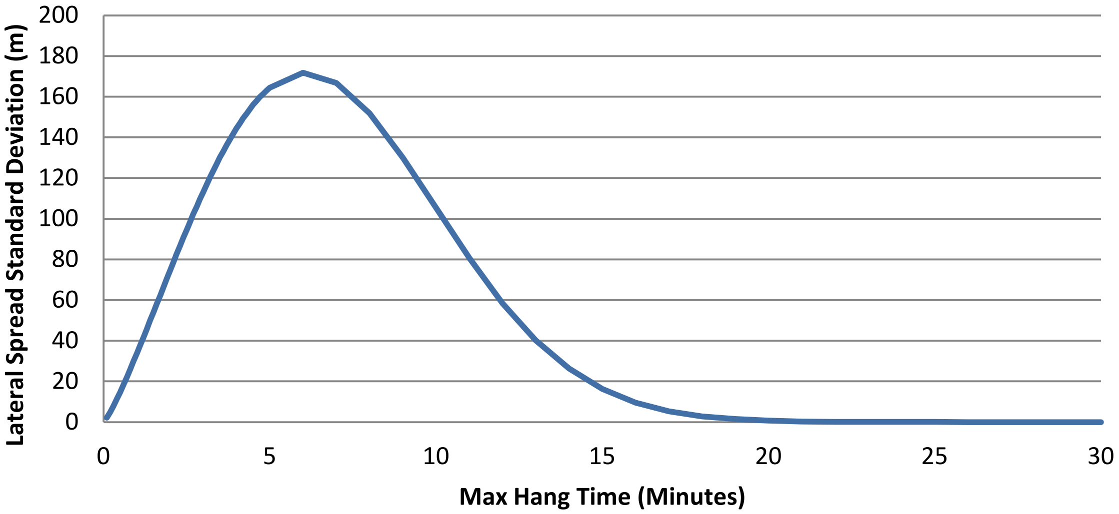

Figure 100. Ember impact lateral spread standard deviation (m) along the ember impact pattern principal axis with an increasing maximum hang time.

The standard deviation value is used in a cumulative normal distribution function do determine the spread and proportion of embers impacting along the central axis. To ensure ember impacts are at a consistent scale to the landscape grid, impacts are calculated at grid cell resolution intervals.

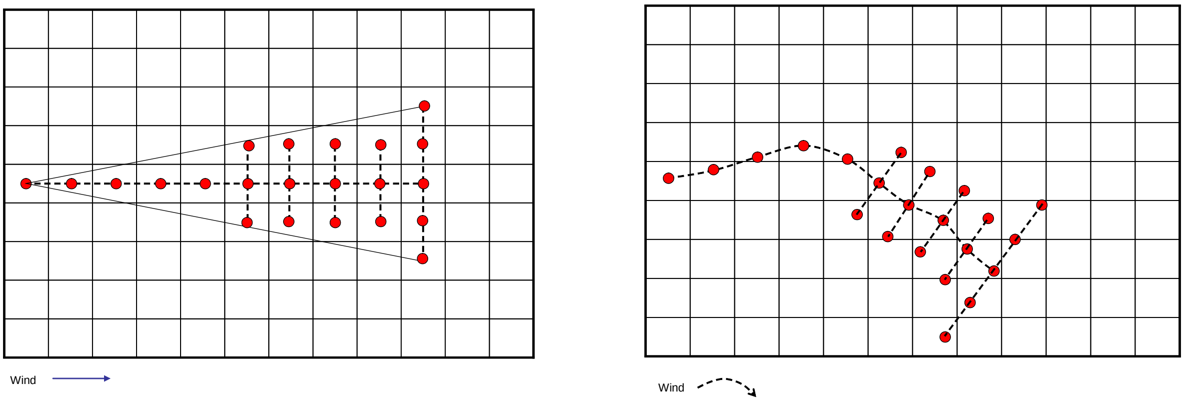

Figure 11. Ember impact patterns show matching grid resolution with a constant (left) and variable (right) wind direction.

Using the prevailing winds, the time to traverse a cell is calculated. Weather inputs are re sampled at every cell traversal to capture any change to direction or speed.

## Spotfire ignition

Modelling every spotfire ignition in the landscape is not computationally feasible. Spotfire ignitions can occur in their millions in the case large fires, most of which are consumed by flaming fronts well before they can establish. The Phoenix spotting model focuses on those impacts far enough downwind from a fire front that effectively become a separate fire large enough to make a significant contribution to a fires progression across a landscape. This distance is assumed to be approximately 200 m.

### *Cumulative ember density*

Following a cell ember launch event, the resulting impact pattern is overlayed on the landscape (Figure 11). The cumulative ember density for each cell is updated based on each impact. A cell can be impacted multiple times from different launch events. The cumulative ember density should not be interpreted as the ‘actual’ ember density affecting a cell, it is a value calibrated to produce an ignition where threshold conditions are met.

### *Spotfire ignition threshold*

A cell spotfire density is calculated based on its cumulative ember density, fuel type, load and corresponding fuel moisture content (FMC).

To do this, ignition probabilities based on a cell’s FMC and fuel load are combined assuming an ember density of 1 ember/m$^2$. A cell’s estimated spotfire density is then calculated by multiplying this probability by its cumulative ember density. Where a cell contains both grass and forest fuel this value is area weighted.

Forest fine fuel moisture content is calculated using the simplified littler layer fuel moisture model developed for eucalyptus forests by CSIRO (Matthews, Gould et al. 2010). The FMC ignition probability function is based on a model for line fire ignitions in Mediterranean grass fuels (Dimitrakopoulos, Mitsopoulos et al. 2010), modified based on the assumption that ember ignitions and forest fuels having a lower surface area to volume ratio will be harder to ignite.

$$Ignition\ Probabilty_F = 0.9 \div \left(1 + e^{-(4.5 - 0.5 \times fmc)}\right)$$

Dead fuel moisture content for grass is determined using the fuel moisture function from the CSIRO grassland fire spread meter (Cheney, Gould et al. 1998). The grass FMC ignition probability is the product of a grass FMC ignition probability function and the grass curing coefficient, used to describe the percentage of cured grass. The grass FMC ignition probability function is similarly based on (Dimitrakopoulos, Mitsopoulos et al. 2010) and modified assuming ember based ignitions will be more difficult than direct flame from a drip torch.

$$Curing_{coeff} = 1.12 \div (1 + 59.2\ \times e^{(-0.124 \times Curing - 50)})$$

$$Ignition\ Probability_{fmc} = 1 \div (1 + e^{-(6 - 0.263\ \times fmc)})$$

$$Ignition\ Probability_G = Curing_{coeff}\ \times Ignition\ Probability_{fmc}$$

The affect of varying fuel loads in t/ha is captured by the following function, with grass loads supplied as a factor of 4.

$$Ignition\ Probability_{FL} = 1 \div (1 + 350 \times e^{-0.55 \times Load})$$

### *Spotfire ignition grid*

Starting a spotfire on every cell that exceeds a spotfire count of 1 is problematic as the landscape grid cell resolution is allowed to be varied. Halving the resolution will result in up to 4 times the number of spotfire ignitions which can drastically affect the final result. To allow grid cell resolution to change whilst maintaining a consistent spotfire ignition resolution a 200 m spotfire ignition grid is used to control the density and locating of the resulting ignitions. Ignitions of spotfires are assumed to occur at the centroid of the 200 m cell.

When a cell impact is logged any intersecting ignition grid cells are identified. An area weighted spotfire density is calculated for each of these ignition grid cells by summing the spotfire densities for each intersected landscape cell. If a cell is partially intersected only the intersected area is used.

When the resulting spotfire density is greater than or equal to one, the ignition grid cell is flagged as ignited and a spotfire added to the landscape at its centroid.

## Conclusions

In PHOENIX, the effect of short-distance spotting (< 200 m) is assumed to be part of the flaming zone rate of spread and is not modelled separately.

The effect of middle to long-distance spotting (1 to 30 km) is modelled explicitly in PHOENIX based on the amount of bark material present, the convective strength in a cell from which the embers are launched, the wind speed and direction, and the fuel quantity and moisture content at the location of ember drop. Dur to the lack of direct quantification of these factors, the spotting model in PHOENIX has been empirically calibrated against spotfire ignitions recorded in fire reconstructions.

Spotfire modelling in PHOENIX is unique. Spotfires have a very significant impact on fire behaviour, including the production of firestorms. Medium to long-distance spotting can effectively increase rates of spread by a factor or two or three.

## References

Chong, D., Tolhurst, K. and Duff, T. (2012). " PHOENIX RapidFire 4.0's Convective Plume Model". Technical Report, Bushfire CRC/University of Melbourne.

Cheney, N., J. Gould, et al. (1998). "Prediction of Fire Spread in Grasslands." International Journal of Wildland Fire **8**(1): 1-13.

Dimitrakopoulos, A. P., I. D. Mitsopoulos, et al. (2010). "Assessing ignition probability and moisture of extinction in a Mediterranean grass fuel." International Journal of Wildland Fire **19**(1): 29-34.

Matthews, S., J. Gould, et al. (2010). "Simple models for predicting dead fuel moisture in eucalyptus forests." International Journal of Wildland Fire **19**(4): 459-467.

Sardoy, N., J. L. Consalvi, et al. (2008). "Numerical study of ground-level distribution of firebrands generated by line fires." Combustion and Flame **154**(3): 478-488.

# Appendix 1

```
Public Sub SpotImpact(ByVal SourceCell As GridCell, ByVal EmberImpact As EmberImpact, ByVal SourceFire
As Fire)
            'Implements an ember impact event on a cell.
            Dim dtStart As Date
            Dim pTargetCell As GridCell = Me.GetValue(EmberImpact)
            Dim pIgnitionIndexs As New List(Of GridIndex)
            Dim pCellIndexes As List(Of GridIndex)
            Dim pIgnition, pCellIndex As GridIndex
            Dim pIntersect, pIgnitonCellExtent As Extent
            Dim intFireId As Integer
            Dim pCell As GridCell
            Dim pCellExtent As Extent
            Dim dblSpotFiresArea As Double

            'prerequsite exit conditions
            If pTargetCell Is Nothing Then Exit Sub 'cell does not exist
            If pTargetCell Is SourceCell Then Exit Sub 'impact fall in source cell
            If EmberImpact.Time >= Me.EndTime Then Exit Sub 'outside range of simulation time

            'Initialise cell
            Me.InitialiseCell(pTargetCell, SourceFire.WeatherService, EmberImpact.Time)

            'Log ember impact, keep counting embers even after burnt
            pTargetCell.LogEvent(EmberImpact)

            'get overlapped igniton cell indexes
            pIgnitionIndexs = pSpotFireIgnitionGrid.GetGridIndex(pTargetCell.Extent)

            For Each pIgnition In pIgnitionIndexs

                If pSpotFireIgnitionGrid.GetValue(pIgnition) = 0 Then 'spotfire threshold not yet met

                    'get ignition cell extent
                    pIgnitonCellExtent = pSpotFireIgnitionGrid.GetCellExtent(pIgnition)

                    'get intersected cells
                    pCellIndexes = Me.GetGridIndex(pIgnitonCellExtent)

                    dblSpotFiresArea = 0

                    For Each pCellIndex In pCellIndexes
                        'calculate area weighted ignition probability
                        pCell = Me.GetValue(pCellIndex)
                        pCellExtent = Me.GetCellExtent(pCellIndex)

                        'get cell intersect extent for area weighting
                        pIntersect = pCellExtent.CreateMinimumExtent(pIgnitonCellExtent)

                        If pCell IsNot Nothing Then

                            If pCell.IsBurnt Then
                                'fire already impacting ignition cell,
                                pSpotFireIgnitionGrid.SetValue(1, pIgnition.Row, pIgnition.Column)
                                dblSpotFiresArea = 0
                                Exit For
                            End If

                            'Initialise cell
                            Me.InitialiseCell(pCell, SourceFire.WeatherService, EmberImpact.Time)

                            'calculate spotfire density by area
                            dblSpotFiresArea += pCell.SpotFireDensity(EmberImpact.Time) *
pIntersect.Area
                        End If

                    Next

                    If dblSpotFiresArea / pIgnitonCellExtent.Area >= 0.005 Then '>= 1 Then

                        'start spotfire
```

```
                        'set fire id
                        intFireId = (pIgnition.Row + 1) * (pIgnition.Column + 1)

                        'set start time
                        dtStart = EmberImpact.Time

                        Me.AddFire(New Fire(intFireId, pIgnitonCellExtent.Centroid, 
SourceFire.Resolution, dtStart, SourceFire.Parent, SourceFire.WeatherService, 
SourceFire.PerimeterExport, pTargetCell))

                        'flag cell as having reached its spotfire ignition threshold
                        pSpotFireIgnitionGrid.SetValue(1, pIgnition.Row, pIgnition.Column)
                    End If

                End If
            Next

            pTargetCell.FireHistory.LastEmberSourceId = SourceCell.Id

        End Sub
```

# Appendix 2

```
Private Sub LaunchEmbers(ByRef SourceCell As GridCell, ByVal SourceFire As Fire, ByVal EmberLaunch As 
EmberLaunch)
            'Implements a cell ember launch event
            Dim pTarget As GridCell
            Dim pImpact As EmberImpact
            Dim dblSpotTime As Double
            Dim pEmberPattern As EmberImpactPattern
            Dim pOrigin As MapPoint = SourceCell.Extent.Centroid

            'Embers already launched or spotting does not exceed cell extent
            If SourceCell.FireHistory.EmbersLaunched Then Exit Sub

            'Flag cell as spotted
            SourceCell.FireHistory.EmbersLaunched = True

            'determine ember impact pattern
            pEmberPattern = EmberLaunch.ImpactPattern(Me.Resolution, SourceCell.WeatherService, 
SourceCell.GetLocalWeather(EmberLaunch.LaunchTime, True))

            If pEmberPattern.Count = 0 Then Exit Sub

            'Apply impact pattern to landscape
            For Each pImpact In pEmberPattern
                Me.SpotImpact(SourceCell, pImpact, SourceFire)
            Next

            If LogMaxSpotExtent AndAlso pEmberPattern.MaxSpot IsNot Nothing AndAlso 
pEmberPattern.MaxSpot.Time <= Me.EndTime Then
                'Add McArthru's maximum spotting distance marker if impact pattern exceeds launch cell

                'get end of ember path point
                pTarget = Me.GetValue(pEmberPattern.MaxSpot)

                'if target cell does not exist or spotting doesn't exceed launch cell
                If pTarget IsNot Nothing AndAlso pTarget IsNot SourceCell Then
                    'log max spot distance and hours since start
                    dblSpotTime = pEmberPattern.MaxSpot.Time.Subtract(Me.StartTime).TotalHours
                    pTarget.LogEvent(SourceFire, pEmberPattern.MaxSpot.Time, dblSpotTime, 
pEmberPattern.MaxSpot.DistanceTravelled)

                    'store max McArthur spotting distance for log
                    dblMcArthurMaxSpotDistance = Max(dblMcArthurMaxSpotDistance, 
pEmberPattern.MaxSpot.DistanceTravelled)

                End If
            End If

        End Sub
```

# Appendix 3

```
Public Function ImpactPattern(ByVal CellResolution As Double, ByRef WeatherService As SpotForecast,
ByVal LaunchConditions As WeatherData) As EmberImpactPattern
            'Generate ember impact pattern
            Dim pPattern As New EmberImpactPattern
            Dim pStartPoint As MapPoint
            Dim pTarget As MapPoint
            Dim dblTotalDist, dblMinutesElapsed, dblTime As Double
            Dim pWeather As WeatherData
            Dim dblHangTime As Double = Me.HangTime
            Dim dtStep As Date
            Dim pImpact As EmberImpact
            Dim intCount As Integer = 0
            Dim pVector, pTangent As Vector
            Dim dblCenterPointCount As Double
            Dim dblCurrentCount, dblPrevCount As Double
            Dim dblEmbersDropped As Double
            Dim dtPrevImpact As Date
            Dim dblCenterRatio, dblStdDev As Double
            Dim dblTotalViableEmbers As Double
            Dim dblResolution As Double = CellResolution
            Dim dblStep As Double = dblResolution
            Dim pMaxSpot As MapPoint
            Dim dblHangTimeRef As Double = 35 'reference maximum hang time, previously 25 minutes but
increased to bring in bimodal split earlier

            'Calculate total ember reaching ground with a temp > 300k based on hang time relative to
            dblTotalViableEmbers = Me.EmbersLaunch * Math.Exp(-9 * (HangTime() / dblHangTimeRef))

            'initialise start point
            pStartPoint = pOrigin

            'initialize distance travelled
            dblTotalDist = 0

            'initialize previous impact time
            dtPrevImpact = dtLaunch

            'set previous count to total embers
            dblPrevCount = dblTotalViableEmbers

            Do
                'claculate ember cloud trajectory

                'get weather for sample point, lag values for temp and rh not required
                pWeather = WeatherService.Weather(dtLaunch.AddMinutes(dblMinutesElapsed), 0)

                If pWeather.SpotWindSpeed > 0 Then 'change back to surface wind speed after FireDST
project, will be the same as surface unless ember transport
                    'Calculate elapsed time to travel a cell resolution given the prevailing windspeed
                    dblTime = dblResolution / (pWeather.SpotWindSpeed * 1000 / 60) 'm/minute)
                    dblStep = dblResolution
                Else
                    dblTime = 1 'drop to 1 minute time step and no distance
                    dblStep = 0
                End If

                'increment elapsed time and calculate impact time
                dblMinutesElapsed += dblTime

                'calculate step time
                dtStep = dtLaunch.AddMinutes(dblMinutesElapsed)

                'Calculate target coordinates based on distance travelled and wind direction
                pTarget = pStartPoint.ResultingPoint(dblStep, pWeather.SpotWindDirection)

                'Increment distance traveled from launch
                dblTotalDist += dblStep

                'Calculate remaining embers for distribution
                dblCurrentCount = dblTotalViableEmbers - (dblTotalViableEmbers *
Me.CumulativeEmberProb(dblMinutesElapsed, dblHangTime))
```

```
                'Calculate the number of embers dropped during elapsed time, swaith area of 1 cell
width by the step distance
                dblEmbersDropped = (dblPrevCount - dblCurrentCount)

                'reset prev impact values
                dblPrevCount = dblCurrentCount
                dtPrevImpact = dtStep

                'calculate standard deviation for lateral ember distribution
                dblStdDev = LateralSpreadStandardDev(dblMinutesElapsed)

                'special case for center point, need to calculate in two halves either side of median
(0)
                dblCenterRatio = GetIntegral(-dblResolution / 2, dblResolution / 2, dblStdDev)

                'calculate center point ember count, based on the center cumulative probability
                dblCenterPointCount = dblCenterRatio * dblEmbersDropped

                'create impact point with ember count for center cell
                pImpact = New EmberImpact(pTarget, dtStep, dblCenterPointCount, dblTotalDist)

                'Embers remain and embers dropped exceed minimum threshold, record impact
                If dblCenterPointCount / dblResolution ^ 2 >= dblMinImpactDensity Then

                    'add central impact
                    pPattern.Add(pImpact)

                    If dblCenterRatio < 0.99 Then
                        'add lateral embers

                        'Calculate directional vector
                        pVector = pStartPoint.ResultantVector(pImpact).Normalise

                        'Log impacts to the right
                        pTangent = pVector.Transform_90degrees_Left
                        AddLateralEmbers(pPattern, dblResolution, pImpact, pTangent, dblStdDev,
dblEmbersDropped)

                        'Log impacts to the right
                        pTangent = pVector.Transform_90degrees_Right
                        AddLateralEmbers(pPattern, dblResolution, pImpact, pTangent, dblStdDev,
dblEmbersDropped)
                    End If
                Else
                    'continue calculating ember decay and impacts until Maximum hang time exceeded
                End If

                'Reset start point
                pStartPoint = pTarget

            Loop Until (dblCurrentCount / dblTotalViableEmbers) <= 0.0005 '99.995% of embers dropped
            'continue modelling trajectory until maximum hang time exceeded

            'record maximum spot if impact occured
            If pImpact IsNot Nothing And dblMaxSpotDistance > 0 Then
                'pPattern.MaxSpot = pImpact
                pWeather = WeatherService.Weather(dtLaunch, 0)
                pMaxSpot = ResultingPoint(pStartPoint, dblMaxSpotDistance, pWeather.WindDirection)
'create a wind line

                pPattern.MaxSpot = New EmberImpact(pMaxSpot, dtStep, 0, dblMaxSpotDistance)

            End If

            Return pPattern

        End Function
```