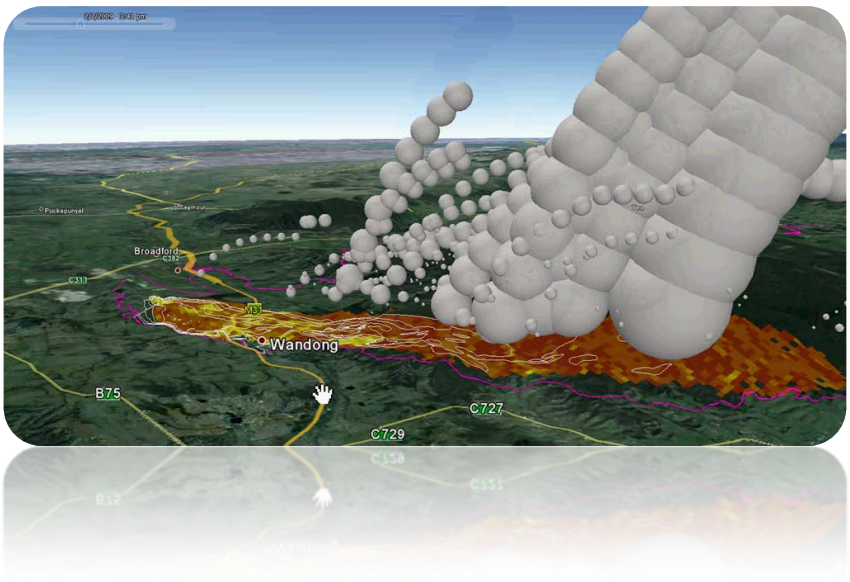
# PHOENIX RapidFire 4.0's Convective Plume Model

## 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:** Representation of the convection in the 2009 Kilmore Fire as produced by PHOENIX RapidFire 4.0

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

Bushfires are a 3-dimensional phenomenon with significant interaction between the surface and the atmosphere. Complex coupled fire-atmosphere models have been produced and give amazingly realistic results, however the computational complexity means that they take many times real-time to run, even on super computers, and are therefore restricted to small areas and short periods of time. PHOENIX RapidFire is primarily a 2-dimensional fire model and only takes a few minutes to run fires in excess of 100,000 ha.

This report describes how PHOENIX RapidFire has been developed to include elements of plume development and ember transport resulting in spot fires. This was done to try and capture some of the important 3-dimensional aspects of bushfires without large computational overheads.

Development of the plume rise and spotting components of PHOENIX has been done with the knowledge of some of the key thermodynamic processes, but a number of assumptions have been made.

Validation of the plume rise and spotting model is difficult because there are no detailed observations recorded for the plume, embers, spotfires and upper-level winds. Ground-based weather radar data was found to be a useful validation dataset for the plume model. The plume model in PHOENIX was calibrated against weather radar data recorded on Black Saturday, 2009.

Early indications are that there has been a significant improvement in the simulation of the Black Saturday fires with the incorporation of the plume and spotting models. Further testing will be required to fully understand the limitations of the model.

# Contents

# Introduction

Plumes or convection columns are a prominent feature of bushfires, and depending on their size and spatial distribution they are often associated with:

- spotting potential
- fluctuations in local wind speed and direction
- complex fire behaviour
- deterioration of air quality

Currently, there are no simple models for bushfire plumes that predict their convective strength or their effects on spotting, air quality or destructive potential.

The PHOENIX bushfire characterization model has made several recent advances in describing the convective elements of bushfires. Efforts to date have focused on identifying surface level, dominant heat centres (convective centres) and using them a predictor of plume locations and strength for ember dispersal. The algorithm performs a surface level aggregation of fire perimeter segments (heat centres) where they are deemed close enough to interact and act as one (Figure 1).

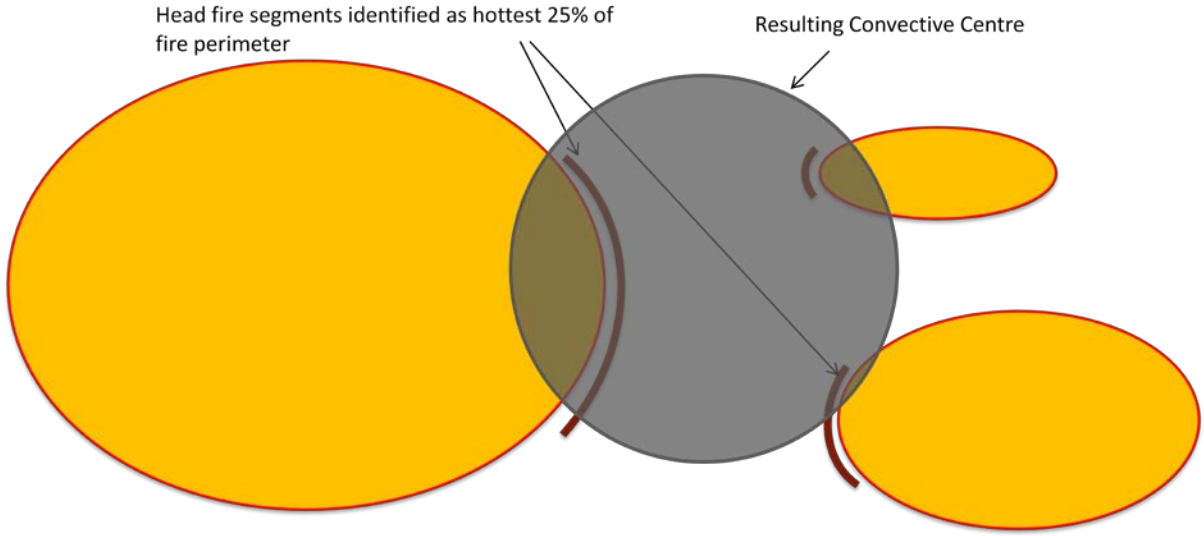



**Figure 1. Fire perimeter 'hottest' segments (brown arcs) are identified then aggregated where appropriate to form a convective centre (grey circle).**

Visual examination of modelled convective centres shows a good match between predicted location and extent and those observed in real bushfires. The heat output and location of these convective areas has been used to provide more realistic results in the PHOENIX ember dispersal and house loss probability model with very promising results.

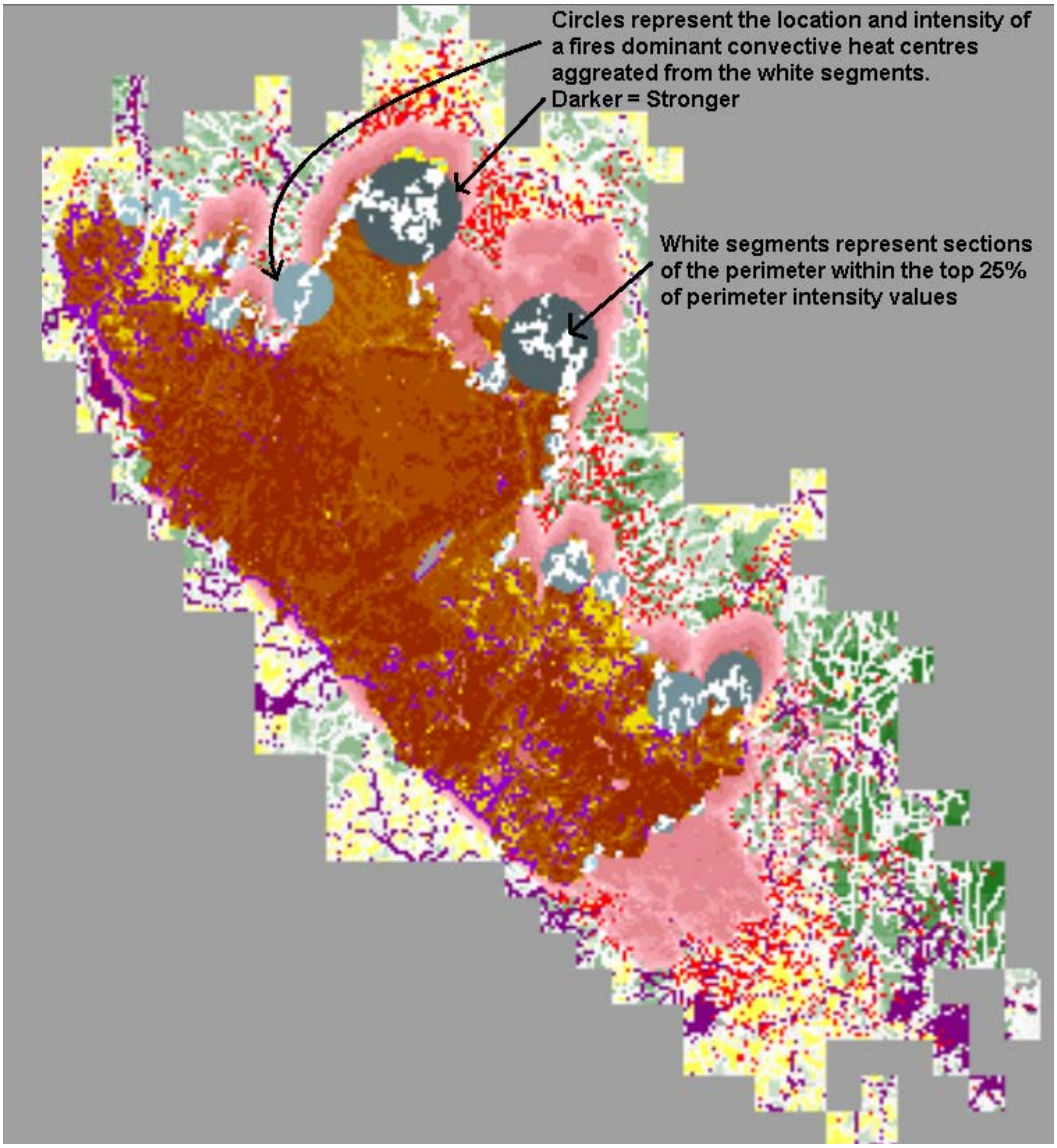



**Figure 2. Image showing heat segments and convective centres from the model run of the 2009 Kilmore fire.**

Whilst a surface level expression of a fire's significant convective centres is useful in its own right, it does not describe the vertical component of the phenomenon. For smoke and ember dispersal; plume height, volume and vertical velocities are important. Over the last 18 months several exploratory investigations into translating these surface convective centres into plumes have been conducted.

It is commonly accepted that ember transport is more a function of winds aloft rather than surface level winds, however, to date; only 10 m input winds are used in operational fire spread prediction. This is a reflection of the both the available forecast and observed data from automatic weather station (AWS), and the difficulty in recording higher level winds as part of experimental fires used to generate the empirical rate of spread functions used in most operational spread models.

Accurate measurements of wind with height and mapping of ember and smoke trajectories is need in order to develop robust defendable plume and ember transport model.

As a plume develops it draws winds down from aloft into its base. Depending on the strength and direction of these winds, the resulting fire behaviour can vary greatly from those expected using 10 m forecast or AWS data. A realistic plume model and accurate vertical wind data may allow equivalent 10 m wind speeds to be determined from upper level winds as plumes develop.

A plume can significantly alter surface level winds around a fire, convective indraughts draw air into the base of a plume affecting the spread rate and direction of surrounding spotfires. A large plume can effectively act as a barrier to ambient wind by entraining it, thus shadowing the areas down wind. It may be possible to incorporate some of these behaviours into PHOENIX by means of a plume model.

A significant amount of work has been carried out in recent years by researchers looking at convective flows around wildfires using coupled atmospheric models (FIRETEC, WRF Fire) and while the work has and continues to provided valuable insights into atmospheric interactions with bushfires, the computational overheads of these models constrain their use to the research domain as computation time is orders of magnitude greater than real time and are also limited in a spatial extents, (generally < 1 square km) due to high input data resolution and temporal resolution. In an operational bushfire prediction context a fire will have probably gone out long before the prediction is complete.

A review of existing operational plume models was conducted and revealed very little was adoptable by the PHOENIX model due to a range of issues including:

- Applicability and scalability in the case of chimney stack base dispersal models, due to column interaction and movement.
- Complexity and performance constraints of running coupled atmospheric/weather based models
- Issues with incorporating moving and variable heat sources

Coupling PHOENIX to an atmospheric model at a suitable scale would result in significant performance overheads and chimney stack based models were too simplistic to accurately model the dynamic nature of bushfire heat emissions at the large bushfire scale.

With limited data available for model development and validation an exploratory green fields approach was taken to developing the plume model. Discussions with Dr Brain Potter (UDSA For. Serv., Pacific NW Research Station) were conducted to identify some broad concepts that would need to be incorporated in order to realistically capture plume dynamics which identified the following as important to the accurate representation of a bushfires plume:

- Represent plume activity around a fires perimeter, not just a single ‘head’ fire plume model. A single plume model fails in the case of large complex fires which can have several active fire fronts and multiple spot fires each generating independent plumes;
- React to changes in wind speed and direction, temperature and relative humidity as the plume rises through the upper atmosphere;
- Capture plume acceleration/deceleration rates;
- Incorporate cooling due to entrainment/mixing;
- Incorporate adiabatic cooling;
- Incorporate latent heat flux due to condensation;

- Model small plumes equally as well as large plumes i.e. scale well.

The model described in this document represents a work in progress, a significant effort is still required to incorporate and validate key elements described above, but results to date are promising.

# The Plume Model

The initial objective for incorporating a plume model into PHOENIX is to allow ember transport winds to be determined by interrogating upper level winds. A simple bubble is used to represent the volume of heated air released from the modelled convective centres in PHOENIX and acts as a tracer for the plume that will form above. As the bubble rises it will be used as the sampling mechanism to determine the ember transport wind speed and direction. Once determined, the terminal velocities of embers in conjunction the plume vertical velocities will be used to calculate ember launch heights.

## Vertical atmosphere

Observation data for plume model development is extremely difficult to obtain. Suitable direct observations are currently limited to weather balloons; however their coarse spatial and temporal resolution (1 or 2 launches a day from a very limited number of locations (2 in Victoria) severely limit the use of this dataset.

Numerical weather models (NWP) can provide a rich picture of the upper atmosphere, but to date NWP data available has come in the form of forecast data which has struggled to match observations at a level that would allow a robust model to be developed (see Chong et al., 2012 for details). This is particularly evident for wind speeds affecting the case study fire areas being considered as part of this project.

For these reasons, incorporating the affects of changes in temperature, relative humidity, wind speed and direction to plume rise will be attempted at a later date as suitable data for model development and validation becomes available.

In order to focus on the vertical elements of a plumes development, a uniformly mixed atmosphere has been assumed with the environmental lapse rate (6.5$^{o}$C drop every 1000 m) assumed.

Again, the availability of accurate observed or reconstructed upper level weather data will be the biggest challenge here, without it, accurate plume model development, calibration and validation will be difficult.

## Initialisation

A representative plume bubble is initialised based on the location, size and strength of each convective centre currently modelled in PHOENIX. These centres represent the areas of convective influence on the ground that surround the base of a plume. In the case of a regular elliptical fire shape this can be defined as the minimum bounding extent containing the hottest 25% of the perimeter (head fire), extended by 110% to capture the area of in draught flows around a plume's base which extend past the burning area.

Plumes are observed to have an almost wine-glass shape, tilted where wind speeds are strong. Air and smoke is drawn in at the base often at quite some distance from the burning area, narrowing above the fire then expanding out as it rises.

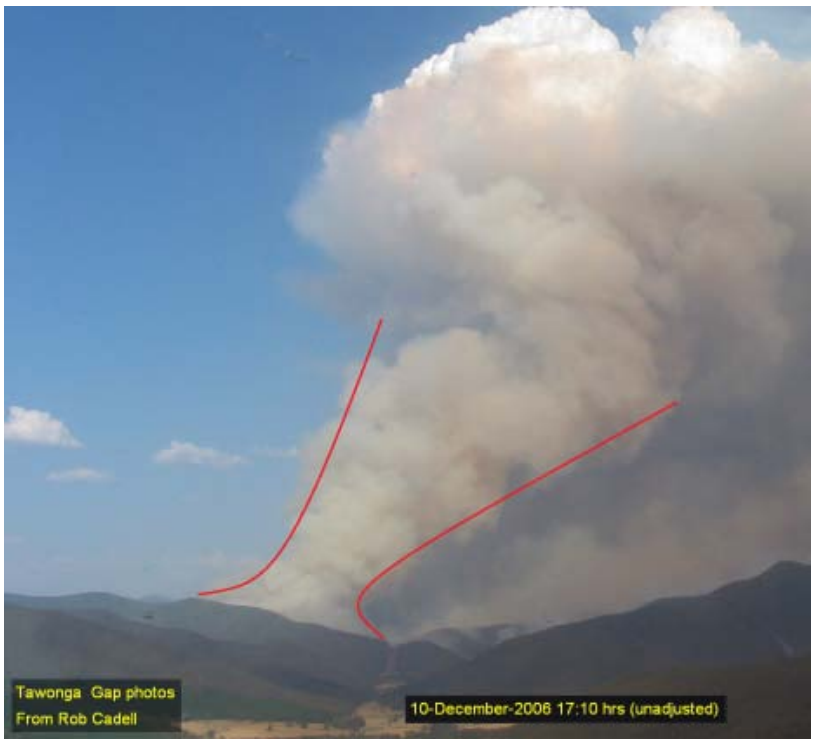



**Figure 3. Classical plume shape showing a wide base narrowing to a convergence zone before expanding out with altitude. (Source: R. Cadell, DSE)**

Plume bubbles are initialised with the radius of narrowest part of a plume where the hot gasses converge above the flaming zone. This is assumed to be 50% of a convective centres effective radius. Initial temperature is determined using Van Wagner's function for estimating convection temperatures above low intensity forest fires (Van Wagner 1975) at a height of 60 m. Plume bubbles are launched at their source convective centres centroid at ground level at the end of each simulation time step.

## Cooling

Two forms of cooling are incorporated into the plume model, cooling due to the mixing of ambient air into the plume and an adiabatic lapse rate. Diffusion is not considered at this time. Temperature rises due to the latent heat of condensation of moisture with the plume is not included at this stage but it has been identified as a significant contributor to plume rise (Potter 2012).

### Laps Rate

A bubble's cooling rate is assumed to follow the dry adiabatic lapse rate of 9.8$^{o}$C per 1,000 m. The environmental lapse rate (ELR) of 6.49$^{o}$C per 1,000 m is assumed for the surrounding atmosphere. The effect of the different lapse rates becomes significant once a bubbles temperature reaches ambient, at which time a bubble will become cooler than the surrounding atmosphere as it continues rising, begin decelerating and eventually descend.

## Entrainment

The entrainment of the surrounding atmosphere into a plume is considered the most significant of the cooling mechanism. The two most significant factors in determining this rate are assumed to be the temperature difference between the plume and the surrounding atmosphere and the perimeter to area ratio at that point. In the case of the plume bubbles, the perimeter to area ratio is replaced by the surface area to volume ratio.

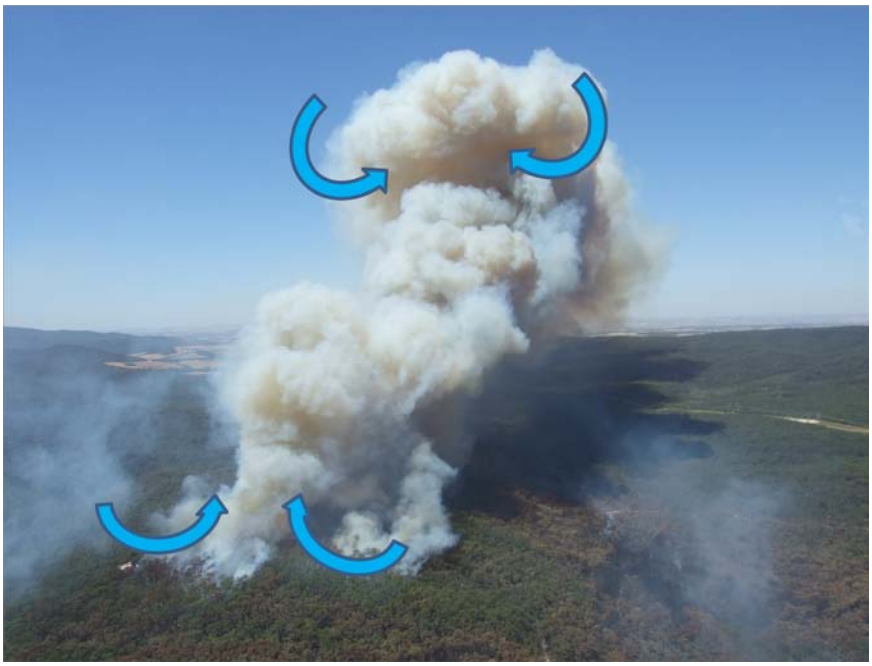


**Figure 4. Entrainment cooling flows represented by blue arrows**

The higher the temperature difference between the plume and ambient air, the more vigorous the mixing process is assumed to be on the plume boundary. The temperature mixing factor $T_f$ is simply expressed as the difference between the bubble temperature $T_b$ and ambient temperature $T_a$.

$$T_f = T_b - T_a$$

The volume of a plume bubble is assumed to be indicative of a plume's volume at a corresponding height.

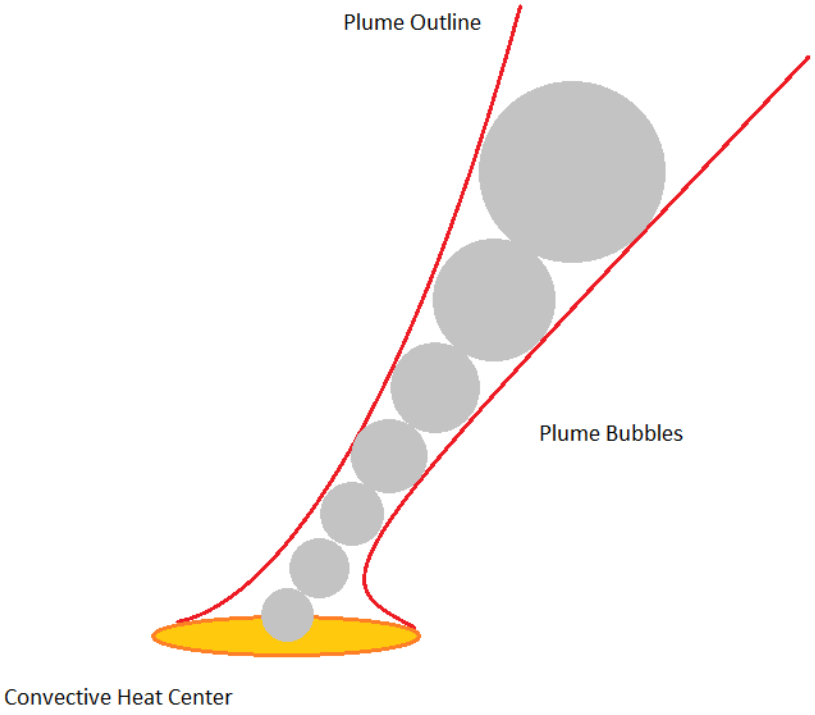



**Figure 5. Plume bubble diameters are assumed to represent the actual plume diameter at the same height.**

The entrainment cooling factor of a bubble $E_{cf}$ is assumed to be inversely proportional to its surface area to volume ratio $R_{sav}$. A smaller bubble will cool more rapidly than a larger bubble at the same starting temperature difference.

$$E_{cf} = 10 \times (R_{sav})^{0.4}$$

A plume bubble's temperature drop $T_d$ is calculated as °C per minute

$$T_d = T_f \times 0.6 \times E_{cf}$$

## Acceleration

It is assumed that plume flows 'generally', accelerate as they approach and enter the base of a plume and continue accelerating as they approach the convergence zone (narrowest part) of a plume. After which they gradually decelerate until they stop climbing (peak height) and start to descend.

Acceleration $A_b$ $(m/s^2)$ in the case of a plume bubble is modelled as a function of bubble density $D_b$ relative to air density $D_a$ at the corresponding.

$$A_b = \frac{9.8 \times (D_a - D_b)}{D_a}$$

## Terminal Velocity

Simply applying unconstrained acceleration to a plume bubble is not realistic, in the absence of a limiting factor bubbles will continue at their maximum speed due to conservation of momentum. Cooling factors will eventually result in deceleration but not at a rate that would account for the flows velocity after the initial acceleration period.

To capture this limiting factor, a terminal velocity is introduced to capture the drag elements that a rising mass or air would experience. It is assumed the bubble will experience unconstrained acceleration until it hits its terminal velocity. It is anticipated that this function will be replaced by a more dynamic drag function to allow actual velocity to be calculated as a bubble rises.

Bubble lift $L_b$ (kg) is calculated as a function of its volume $V_b$ and relative density $D_a - D_b$

$$L_b = V_b \times (D_a - D_b)$$

Bubble vertical velocity ($V, m/s$) is then calculated using the formula below with a drag co efficient $C_d$ of 0.5, the bubble cross sectional area $A_\times$ and ambient air density$D_a$.

$$V = 0.25 \times \left(\frac{L_b \times 9.8}{C_d \times A_\times \times D_a}\right)^{.5}$$

Note the 0.25 value is a scaling factor used to calibrate the plume rise rate against observations in the development of the model.

## Expansion

Plumes increase in size as they rise due to ambient air entrainment, the change in bubble volume $V_b$ to reflect this is modelled as the square of the bubbles temperature change ratio in Kelvin, with $T_p$ being the previous bubble temperature and $T_c$ the current bubble temperature.

$$V_b = V_b \times \left(\frac{T_p}{T_c}\right)^2$$

## Time step scaling (dynamic time steps)

Processing and rendering plume bubbles quickly becomes unmanageable with a fixed time step. PHOENIX incorporates spotting which can quickly generate hundreds of independent fires which can all produce their own plumes. Small plumes require a finer time step to capture rapid cooling rates compared to larger, slower cooling plumes which makes using a fixed time step problematic. With bubbles being relaunched at every time step for significant convective centres, and their trajectories recorded, processing time, computer memory use and graphical rendering of the plume become a major limitation.

Several methods were evaluated to reduce the time steps required to model bubble trajectories to reduce this overhead. The most successful was to set the time step to coincide with 20 evenly spaced intervals along the bubbles vertical travel distance.

This vertical travel distance or maximum height was found to be strongly correlated to the bubbles initial volume to surface area ratio. The variable time step is set as the time required to rise 1/20 of this maximum height given a bubbles current vertical velocity and acceleration.

Results compared favourably to fixed one minute time steps whilst significantly reducing processing requirements. Dynamics of smaller, faster cooling plumes are captured, as are larger slower cooling plumes as the time step is proportional to the plumes maximum height which reflects the rate of change of plume temperatures.

## Plume angle

There are many factors that affect a plumes angle such as plume buoyancy, the fires spread rate relative to wind speed and upper level wind speeds. For this generalized plume implementation surface (10 m) wind speed and direction is used for horizontal bubble transport/deflection of the plume.

## Validation

PHOENIX is currently a surface spread model with no atmospheric coupling. In order to achieve the correct spatial distribution of heat, PHOENIX must initially match the progression the fire accurately. This is achieved by 'fitting' a weather stream to ensure the modelled fire adequately matches the mapped surface spread to ensure a comparable heat distribution. Only the wind direction and weather timing has been adjusted in these validation cases. Timings are adjusted based on the distance and direction of the head of the fire to the AWS location used for the observations data.

BOM Radar data is currently the only reliable data suitable for calibration and validation of the plume model. With the fires of 7 Feb 2009 in Victoria being the best documented in terms of surface spread. Unfortunately the most relevant radar installation failed for a period of approximately 4 hours (14:30 – 18:20) which was the period of the major run for the two largest fires of the day. The 'available' Radar scan line data has been animated in GoogleEarth and used to visually calibrate and validate the plume model.

Bushfire dynamics are extremely complex with atmospheric coupling becoming significant as fires increase in size and heat output. Data required to develop and validate models that capture these dynamics is extremely limited, even more so for large bushfires. The primary focus on large, destructive fires is protection of life and property, the collection of scientific data suitable for model development and validation is not a priority it these cases and would likely interfere. The availability of quality data is the biggest limitation to the development of a robust plume model in Phoenix.

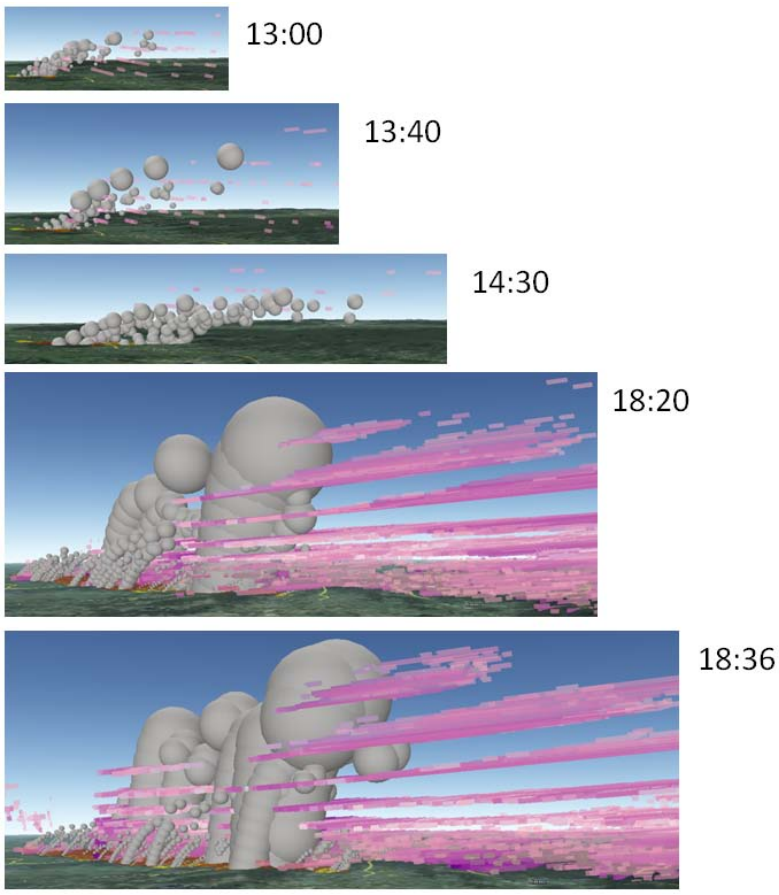



**Figure 6.** South-easterly run of the Kilmore fire showing modelled plume bubbles as white spheres compared to radar reflectance scans indicating smoke location and density in pink.

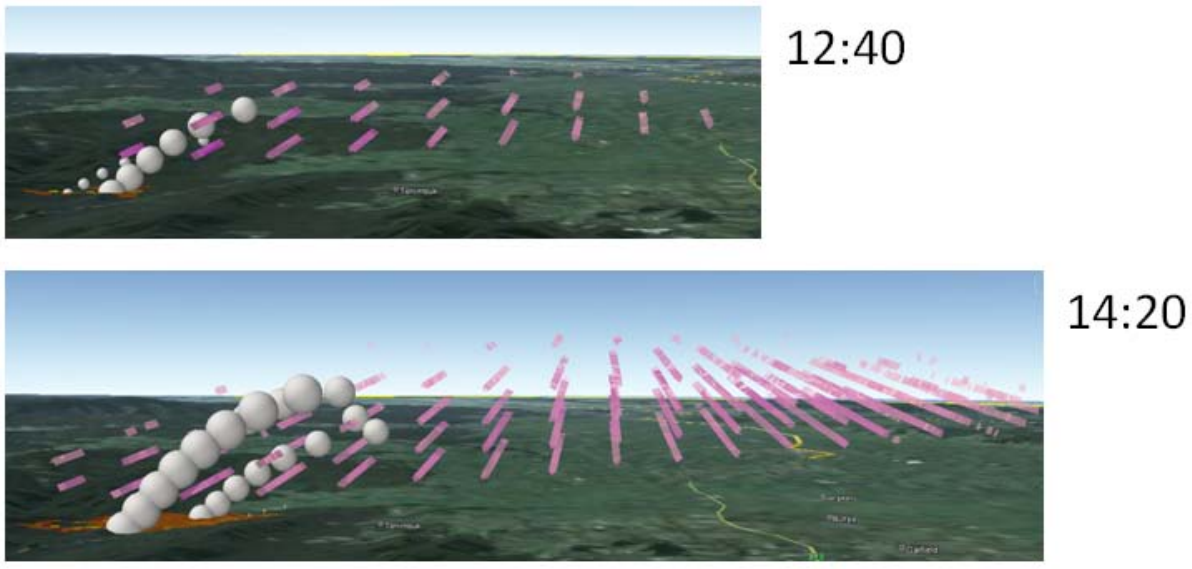



**Figure 7.** Two comparison images of the Bunyip Ridge fire in its initial 2 hours after breaking out. Post wind change comparisons are not possible as the smoke from the Kilmore Fire obscures the plume.

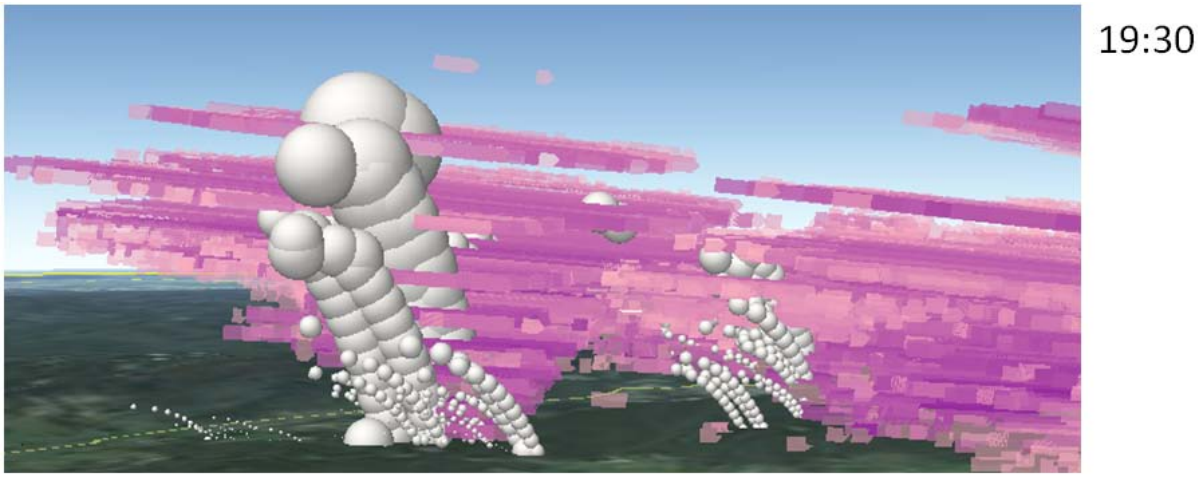



**Figure 8. Murrindindi fire comparison shortly after the wind change. Smoke on the far right is form the Kilmore fire.**

## Conclusions

Whilst the PHOENIX plume model shows promising results it is important to recognise its underlying functions are based on a simplified upper atmosphere and fitted to events of a single day. The biggest challenges to validating and improving the model are the availability of accurate:

- surface and upper atmospheric data
- fire progression reconstructions
- observations of plume development and dynamics

Collating and processing these datasets is a complex and time consuming exercise with little in the ways of standards or tools to guarantee consistency. Given these limitations, any coupling of the plumes modelled in PHOENIX 4.0 with other fire spread mechanism within PHOENIX is not supported. The current implementation in PHOENIX 4.0 should be used for indicative purposes only.

For the FIRE-DST case-studies undertaken for the Bushfire CRC, retrospectively generated forecast grids supplied by the BOM provide an amazing insight into upper atmospheric weather patterns of these severe fire weather days. However, they exhibit significant biases, especially in wind speed, which extend from surface to higher levels limiting their use in plume model development and validation (see Chong et al 2012 for details).

Apart from the Victorian fires of the 7$^{th}$ February 2009, case study fire progression reconstructions are highly stylised and lack sufficient detail to be confident of the fires location and propagation mechanisms (spotting VS surface) at particular times.

Reliable radar data for validating plume dynamics is limited to only a few locations nationally and processing this proprietary data into a form usable for validation requires specialist tools and skills to be developed.

Literature on plume dynamics of large bushfires show that to date, studies have been largely theoretical or limited by scale in the case of coupled atmospheric modelling. This has restricted the option of adopting 'readily available' models for plume modelling within PHOENIX. The literature

however, reveals an increasing recognition of the importance of convective plumes and atmospheric coupling on fire spread and in particular the spotting phenomenon.

## References


Chong, D., Tolhurst, K. and Duff, T. (2012). "Incorporating vertical winds into PHOENIX RapidFire's ember dispersal model". Technical Report, Bushfire CRC/University of Melbourne.

Potter, B. E. (2012). "Atmospheric interactions with wildland fire behaviour – II. Plume and vortex dynamics." International Journal of Wildland Fire **21**(7): 802-817.

Van Wagner, C. E. (1975). "Convection temperatures above low intensity forest fires. Can. For. Serv. Bi-mon. Res. Notes." Can. For. Serv. Bi-mon. Res. Notes **31**(2): 21.

# Appendix 1

```
Public Class Bubble
        'Bubble implementation of a covective plume
        Implements IComparable(Of Bubble)

        Private colHistory As New List(Of BubbleState)
        Private pState As BubbleState
        Private dblDensity As Double 'kg/m3
        Private dblDragCoeff As Double
        Private pSpotForecast As SpotForecast
        Private blnExclude As Boolean = False
        Private dblMinutes As Double
        Private blnDisplay As Boolean = False

        Public Sub New(ByVal CurrentTime As Date, ByVal Temperature As Double, ByVal Radius As Double,
ByVal Elevation As Double, ByVal X As Double, ByVal Y As Double, ByVal Weather As SpotForecast)
            Dim pWeather As WeatherData

            pSpotForecast = Weather
            pWeather = pSpotForecast.Weather(CurrentTime, 0)

            'create new bubble state and add to collection
            pState = New BubbleState(X, Y, Elevation, Radius, CurrentTime, Temperature, 0.1) 'start
with small vertical velocity
            colHistory.Add(pState.Clone)
        End Sub

        Public Property Display As Boolean
            Get
                Return blnDisplay
            End Get
            Set(value As Boolean)
                blnDisplay = value
            End Set
        End Property
        Public Function Merge(ByVal Bubble As Bubble) As Boolean
            'returns true if bubbles merged
            Dim dblTotalVolume As Double

            If Not WillMerge(Bubble) Then Return False

            'merge bubble properties
            dblTotalVolume = Me.State.Volume + Bubble.State.Volume

            'calculate volume weighted bubble temperature
            Me.State.Temperature = ((Me.State.Temperature * Me.State.Volume) + (Me.State.Temperature *
Bubble.State.Volume)) / dblTotalVolume
            Me.State.Volume = dblTotalVolume
            Me.colHistory.AddRange(Bubble.colHistory)

            Return True

        End Function

        Private Function WillMerge(ByVal Bubble As Bubble) As Boolean
            Dim dblCentersDistance As Double
            'Return False

            dblCentersDistance = Me.State.Location.Distance(Bubble.State.Location)

            If dblCentersDistance < Me.State.Radius + Bubble.State.Radius Then
                Return True
            Else
                Return False
            End If

        End Function
        Property State As BubbleState
            Get
                Return pState
            End Get
            Set(value As BubbleState)
```

```
                pState = value
            End Set
        End Property
        ReadOnly Property History() As List(Of BubbleState)
            Get
                Return colHistory
            End Get
        End Property
        Public Function MaxHeight(VolumeToSurfaceArea As Double) As Double
            'Approximation of maximum plume height based on surface area to volume ratio, needs to
incorporate temperature in the future
            ' coefficients
            Const a As Double = -18108.8075045193
            Const b As Double = 6423.57301037272
            Const c As Double = 0.109667927227212
            Const d As Double = 17.7860120628726

            'reduce by .6 to approximate lowest temperature values
            VolumeToSurfaceArea = VolumeToSurfaceArea * 0.6

            Return a + b * Math.Log(c * VolumeToSurfaceArea + d)

        End Function
        Public Sub TimeStep(ByVal TargetTime As Date)
            'model bubble rise to the supplie time
            Dim dbltimestep As Double
            Dim dblVolSA As Double = pState.Volume / pState.SurfaceArea
            Dim dblTemp As Double = pState.Temperature
            Dim dblResolutionTimeStep As Double
            Dim dblMaxHeight As Double

            If pState.Height > 20000 Then Exit Sub 'don't perform any convection modelling above 20km
ceiling

            If blnExclude Then Exit Sub

            'approximate max height
            dblMaxHeight = MaxHeight(pState.Volume / pState.SurfaceArea)

            Do
                If colHistory.Count < 2 Then 'first time steps, make it small to initialise
acceleration
                    dblResolutionTimeStep = 0.2
                Else
                    'calculate time step to cover sample distance
                    dblResolutionTimeStep = pState.TimeStep(dblMaxHeight / 20) 'sample distance based
on approximate max height, 20 sample points
                End If

                'dblResolutionTimeStep = 0.1

                'calculate remaining timestep
                dbltimestep = TargetTime.Subtract(pState.CurrentAt).TotalMinutes

                If dbltimestep < dblResolutionTimeStep Then
                    dblResolutionTimeStep = dbltimestep
                End If

                'perform timestep
                Me.Increment(dblResolutionTimeStep)

                colHistory.Add(pState.Clone) 'add new state to history

                If pState.VerticalVelocity <= 1 AndAlso pState.CurrentAcceleration <= 0.1 Then
                    blnExclude = True
                ElseIf pState.Height > 20000 Then 'do not go above 20 km
                    blnExclude = True
                End If

            Loop Until pState.CurrentAt = TargetTime Or blnExclude  'model until end of time or when
bubble stops rising and has been excluded

        End Sub
```

```
        Private Sub Increment(ByVal Minutes As Double)
            'model bubble movement for duration
            Dim dblAirTemperature, dblHorizontalDistance As Double
            Dim pWeather As WeatherData

            pWeather = pSpotForecast.Weather(Me.State.CurrentAt, 0)
            dblAirTemperature = pWeather.Temperature - (0.0065 * pState.Height) 'caluclate air
temperature at balloon altitude using environmental laps rate

            'update bubble state
            pState.Update(dblAirTemperature, Minutes)

            pState.HorizontalVelocity = pWeather.WindSpeed / 3.6 'convert to m/s

            dblHorizontalDistance = pWeather.WindSpeed * 1000 * (Minutes / 60) 'horizontal distance
travelled in m

            'calculate new location based on wind speed and direction
            pState.Location = pState.Location.ResultingPoint(dblHorizontalDistance,
pWeather.WindDirection)

        End Sub

        Public Function CompareTo(other As Bubble) As Integer Implements System.IComparable(Of
Bubble).CompareTo
            'reverse sort in decending order
            If Me.pState.Radius > other.pState.Radius Then
                Return -1
            ElseIf Me.pState.Radius < other.pState.Radius Then
                Return 1
            Else
                Return 0
            End If
        End Function

    End Class
```

# Appendix 2

```
Public Class BubbleState
        'Captures a bubbles current state and manages transition to next time increment
        Private dblX, dblY, dblHeight, dblRadius As Double
        Private dblVerticalVelocity, dblHorizontalVelocity As Double 'm/s
        Private dblCurrentAcceleration As Double 'm/s
        Private dblCurrentTerminalVelocity As Double 'm/s
        Private dblTemperature As Double 'C
        Private dtCurrentAt As Date
        Private dblMinutes As Double
        Public dblMaxVerticalVelocity As Double
        Public Sub New(ByVal X As Double, ByVal Y As Double, ByVal Height As Double, ByVal Radius As
Double, ByVal CurrentAt As Date, ByVal Temperature As Double, ByVal VerticalVelocity As Double)
            dblX = X
            dblY = Y
            dblHeight = Height
            dblRadius = Radius
            dblTemperature = Temperature
            dtCurrentAt = CurrentAt
            dblVerticalVelocity = VerticalVelocity
        End Sub
        Public Function TimeStep(ByVal SampleDistance As Double) As Double
            'determine the next time step required to meet the distance increment
            Dim dblATimeStep As Double

            If Me.CurrentAcceleration > 0 Then
                dblATimeStep = (-Me.VerticalVelocity + Math.Sqrt(Me.VerticalVelocity ^ 2 - 4 * 0.5 *
Me.CurrentAcceleration * -SampleDistance)) / (2 * 0.5 * Me.CurrentAcceleration)
            Else
                dblATimeStep = SampleDistance / Me.VerticalVelocity
            End If

            Return dblATimeStep / 60 ' convert to minutes

        End Function

        Property Temperature As Double
            Get
                Return dblTemperature
            End Get
            Set(value As Double)
                dblTemperature = value
            End Set
        End Property
        Property Location As MapPoint
            Get
                Return New MapPoint(X, Y)
            End Get
            Set(value As MapPoint)
                dblX = value.X
                dblY = value.Y
            End Set
        End Property
        Property VerticalVelocity As Double 'm/s
            Get
                Return dblVerticalVelocity
            End Get
            Set(value As Double)
                dblVerticalVelocity = value
            End Set
        End Property
        ReadOnly Property CurrentTerminalVelocity As Double 'm/s
            Get
                Return dblCurrentTerminalVelocity
            End Get
        End Property
        ReadOnly Property CurrentAcceleration As Double 'm/s2
            Get
                Return dblCurrentAcceleration
            End Get
        End Property
        Property HorizontalVelocity As Double 'm/s
```

```
            Get
                Return dblHorizontalVelocity
            End Get
            Set(value As Double)
                dblHorizontalVelocity = value
            End Set
        End Property
        Property X As Double
            Get
                Return dblX
            End Get
            Set(value As Double)
                dblX = value
            End Set
        End Property

        Property Y As Double
            Get
                Return dblY
            End Get
            Set(value As Double)
                dblY = value
            End Set
        End Property

        Property Height As Double
            Get
                Return dblHeight
            End Get
            Set(value As Double)
                dblHeight = value
            End Set
        End Property
        Public Property Volume As Double
            Get
                Return (4 / 3) * Math.PI * Me.Radius ^ 3
            End Get
            Set(value As Double)
                dblRadius = VolumeToRadius(value)
            End Set
        End Property
        Public ReadOnly Property SurfaceArea As Double
            Get
                Return 4 * Math.PI * Me.Radius ^ 2
            End Get
        End Property

        Private Function VolumeToRadius(ByVal Volume As Double) As Double
            Return ((3 * Volume) / (4 * Math.PI)) ^ (1 / 3)
        End Function

        Property Radius As Double
            Get
                Return dblRadius
            End Get
            Set(value As Double)
                dblRadius = value
            End Set
        End Property

        Property CurrentAt As Date
            Get
                Return dtCurrentAt
            End Get
            Set(value As Date)
                dtCurrentAt = value
            End Set
        End Property
        Private Function TerminalVelocity(ByVal AirTemperature As Double) As Double
            'calculates bubble terminal velocity based on difference in air density.
            'and bubble cross sectional area
            'bigger bubbles should have more drag but ^3 volume relationship to radius should
compensate in terms of lift.
            Dim dblVelocity As Double
```

```
            Dim dblAirDensity, dblBubbleDensity As Double
            Dim dblAirPressure As Double
            Dim dblDragCoefficient, dblCrossArea As Double

            'calculate air pressure and temperature at bubble height
            dblAirPressure = 1000 * WeatherData.AltitudePressureKpa(Me.Height) 'pressure in pa

            'calculate densities in kg/m3
            dblAirDensity = dblAirPressure / (287.05 * (AirTemperature + 273.15))
            dblBubbleDensity = dblAirPressure / (287.05 * (Me.Temperature + 273.15))

            Dim dblLift As Double = Me.Volume * (dblAirDensity - dblBubbleDensity)

            dblLift = Math.Max(dblLift, 0) 'fence at positive lift.

            dblDragCoefficient = 0.5

            dblCrossArea = Math.PI * Me.Radius ^ 2

            dblVelocity = Math.Sqrt(dblLift * 9.8 / (0.5 * dblDragCoefficient * dblCrossArea *
dblAirDensity)) / 4 'reduce by a factor of 4 to keep below 200 km/h

            Return dblVelocity
        End Function

        Private Function Acceleration(ByVal BubbleTemperature As Double, ByVal AirTemperature As
Double, ByVal Altitude As Double) As Double
            'returns acceleration in m/s^2
            Dim dblAccleration As Double
            Dim dblAirDensity, dblBubbleDensity As Double
            Dim dblAirPressure As Double

            'calculate air pressure and temperature at bubble height
            dblAirPressure = 1000 * WeatherData.AltitudePressureKpa(Altitude) 'pressure in pa

            'calculate densities
            dblAirDensity = dblAirPressure / (287.05 * (AirTemperature + 273.15))
            dblBubbleDensity = dblAirPressure / (287.05 * (BubbleTemperature + 273.15))

            dblAccleration = 9.8 * (dblAirDensity - dblBubbleDensity) / dblAirDensity

            Return dblAccleration

        End Function

        Public Function Update(ByVal AirTemperature As Double, ByVal Minutes As Double) As Boolean
            'calculates the bubble temperature drop for a given time
            Dim dblCoolingFactor, dblTemperatureDrop As Double
            Dim dblTempDiff As Double
            Dim dblDistance As Double
            Dim dblNewVelocity As Double
            Dim dblInitialTemperature As Double
            Dim dblAcceleration As Double

            'increment total minutes
            dblMinutes += Minutes

            dblInitialTemperature = Me.Temperature

            'set current acceleration
            dblAcceleration = Acceleration(Me.Temperature, AirTemperature, Me.Height)


            'calculate new velocity based on acceleration, ignoring drag
            dblNewVelocity = (Me.VerticalVelocity + (Me.VerticalVelocity + (dblAcceleration * (Minutes
* 60)))) / 2

            'set current terminal velocity
            dblCurrentTerminalVelocity = Me.TerminalVelocity(AirTemperature)

            'limit bubble assent rate to terminal velocity
            If dblNewVelocity > Me.CurrentTerminalVelocity Then
                dblNewVelocity = Me.CurrentTerminalVelocity
            End If
```

```
            dblMaxVerticalVelocity = Max(dblMaxVerticalVelocity, dblNewVelocity)
            'calculate temperature difference
            dblTempDiff = Me.Temperature - AirTemperature

            '1 minute entrainment factor, larger bubbles take longer to cool due to surface area to
volume ratio
            'Ambient air takes longer to mix in
            dblCoolingFactor = 10 * (Me.SurfaceArea / Me.Volume) ^ 0.4

            'assume 60% change in temperature difference per minute
            dblTemperatureDrop = dblTempDiff * 0.6 * dblCoolingFactor * Minutes

            If dblTemperatureDrop >= dblTempDiff Then
                'drop due to entrainment/mixing cannot go below air temperature, set bubble
temperature to air temperature
                Me.Temperature = AirTemperature
            Else
                Me.Temperature = (Me.Temperature + (Me.Temperature - dblTemperatureDrop)) / 2
            End If

            'update current accleration based on change in velocity
            dblCurrentAcceleration = (dblNewVelocity - Me.VerticalVelocity) / (Minutes * 60)

            Me.VerticalVelocity = dblNewVelocity

            'update state for height change
            dblDistance = Me.VerticalVelocity * (Minutes * 60)

            Me.Height = Me.Height + dblDistance

            Me.Temperature -= dblDistance * 0.01 'apply environmental lapse rate to baloon temperature

            'recalculate volume, assume inversely porprotional to temperature drop
            Me.Volume = Me.Volume * ((dblInitialTemperature + 273.15) / (Me.Temperature + 273.15)) ^ 2

            Me.CurrentAt = Me.CurrentAt.AddMinutes(Minutes)

        End Function

        Public Function Clone() As BubbleState
            Return New BubbleState(Me.X, Me.Y, Me.Height, Me.Radius, Me.CurrentAt, Me.Temperature,
Me.VerticalVelocity)
        End Function

    End Class
```