Bachelor thesis, 15 higher education credits, C-level

BANG, A SYSTEM FOR SURVEYING THE SEPARATION POINTS OF CARGO GRENADES

Benjamin Donald Oakes

Audio engineering programme, 180 higher education credits

Örebro, spring term 2011

Examiner: Dag Stranneby

Örebro universitet
Akademin för naturvetenskap och teknik
701 82 Örebro

Örebro University
School of Science and Technology
SE-701 82 Örebro, Sweden

Summary

This report describes the theory of the surveying system BANG which is used to determine the position and time point of one to two separations of cargo grenades. From the time differences of microphone registrations and microphone positions, the sought coordinates and time points of a separation can be determined.

The major aim of the current project was to recode an earlier version of the software program written in FORTRAN 77 to a modern high level language. This report describes the theory and calculations made by the new program BANG, practical use of the new software program, an alternative graphical solution which ensures convergence and in addition improvements and expansions of the software. A few words on the choice of microphone is also included.

Summary, Swedish

Rapporten beskriver teorin om positioneringssystemet BANG som används för att avgöra positionen och tidpunkten hos en till två separationer av cargogranater. Utifrån tidsskillnader hos mikrofonregistreringar samt kännedom om mikrofonernas positioner, kan de sökta koordinaterna och tidpunkterna beräknas.

Målet med examensarbetet var att koda om den befintliga mjukvaran av programmet skriven i FORTRAN 77 till ett modernt högnivåspråk. Rapporten beskriver teorin och beräkningar utförda av det nya programmet, praktisk användning av det nya mjukvaruprogrammet, en ny alternativ metod som använder sig av en grafisk lösning för garanterad konvergens samt möjligheter till förbättringar och påbyggnader. Några ord om mikrofonval yttras i slutet av rapporten.

Foreword

I would like to acknowledge

Dag Stranneby, ÖU

For helping me find and realise this fantastic graduate job. He is an inspiration and a fountain of knowledge.

Carl Arnesson and **Tord Kemi**, Bofors Test Center

For the graduate job.

Niklas Gillström, Student ÖU

For helping me with bug management and integrating Matlab in C# for graphical view of result.

Timmy Jahrl, Student ÖU

For helping me with matrix methods using base vectors for converting to another coordinate system.

And finally I'd like to thank my supervisor at ÖU

Kjell Mårdensjö, ÖU

Worthy of a gold medal for his good advice, inspiration and guidance into the world programming technology.

Table of contents

Background

BANG developed by Bofors Test Center, is a system for determining the position and time point of one or two separations of certain projectiles. The purpose of this system is to survey the separation point of cargo grenades.

In the situation of one bang, the coordinates and time point of separation are calculated directly with BANG's main algorithm, based on minimisation of a cost function. When more than one bang is present, each bang is discriminated first by finding the smallest sum of residuals out of all possible combinations of registration times. Once the microphone registrations have been assigned to each bang, more accurate approximations are then obtained for each bang using cost function minimisation as in the case of a single bang. This algorithm is also required to distinguish a sonic boom which can be generated by the shock wave as the grenades travel at supersonic velocity.

The previous software program BANG was last updated in 1991. At the time this project was started, the software program BANG was an MS-DOS application coded in FORTRAN 77. It was difficult to use and poorly documented.

Objectives of this work

The primary objective of this work was to make the software of BANG more user friendly and document the code well.

Requirements

Primary requirements:

1. Rewrite the software program in a modern high level language, preferably C#.
2. The program should be compatible with Windows XP, Vista and Windows 7

Secondary requirements:

1. Input data should be easy to feed in, both manually and by reading text files.
2. The result should be simple to interpret and be written to text files.

Stretch goals:

1. Develop the functionality of the program. A few points:
   1. Can more than two bangs be discriminated at once?
   2. Develop weather data calculations so they are more accurate.
2. Can separation be done with frequency analysis?
3. Smaller study of the hardware involved. Do microphones need to be changed?
4. Are there other possible algorithms better than the currently used method? Analyse!

The theory used in BANG is described in the report. Documentation is important for continued development. The code is documented with Doxygen, an html documentation.

Definitions

Here are some definitions of terms frequently used in this report.
Note: if the definition has a unit, the unit is stated after the explaining text.

$bang$ – sound from the separation of a cargo grenade, can also refer to a loud noise or sonic boom

$nbang$ – number of bangs

$nmic$ – number of microphones

$xmic(n), ymic(n), zmic(n)$ – coordinates for the position of microphone n *(m)*

$tmic(n,k)$ – the scaled registration time of bang k at microphone n *(m)*

$x0, y0, z0$ – start guess of the position of one bang *(m)*

$t0$ – start guess of the first registration time *(m)*

xb, yb, zb, tb – currently estimated position and first registration time for a bang *(m)*

$tbtrue$ – true time of first microphone registration *(s)*

inc – step length for numerical differentiation, 0,0001 good enough

$maxiter$ – maximal number of iterations for Newton-Raphson

dz – integration step in height (from microphone to bang) *(m)*

ds – integration step along true line (from microphone to bang) *(m)*

$nstep$ – Number of integration steps $= \left(\frac{zb-zmic(n)}{dz}\right)int$

$aspeed$ – reference speed of sound ≈ 340,3 *(m/s)*

$ss(i)$ – local speed of sound at step i *(m/s)*

q – cost function value *(m^2)*

$residual$ – the smallest value of q found for a tried combination of times *(m^2)*

Note: all the listed times here are scaled to distances with the speed of sound, initially $aspeed$ to simplify calculations.

Markings:

( ) – figures

[ ] – expressions, equations

{ } - references

Theory description/method (for the new BANG)

For re writing the old FORTRAN 77 code to an object oriented C# application with optimal solutions, an understanding of the structure and theory of the old BANG program was required. This part of the work goes beyond a straight "translation" from FORTRAN to C#. The C# structure is very different to FORTRAN and improved. In order to improve the efficiency and functionality of the program an understanding of the theory behind BANG was necessary. This was needed not only to improve the program but also because vital code in the old BANG was written in machine code, making it difficult to "blindly copy" the code. Identifying and analysing the underlying mathematical methods was a major part of this work.

Overview

A number of microphones are placed about the field of fire. The microphones are sampled at a relatively high frequency, 40 kHz. The sample data from each microphone is saved in text files (IRIG-B, GPS time code). The registration times of each microphone are found by checking at which sample the voltage exceeds a certain threshold. The sample number is multiplied by the sample time 1/40 k = 25 µs to obtain the time.
From the time data of each microphone and knowledge of the positions of the microphones, the position and time point of a bang in the air can be determined. The principle is analogous of GPS-positioning, only using sound instead of electromagnetic radiation.

How BANG works

Because of messy algebraic equations (see page 12), imprecise measurements and weather distortion from the real world, BANG uses an advanced method of optimisation with Newton-Raphson at the bottom.

Calculations in BANG:

The engine or ”motor” in BANG consists essentially of two parts:

1. Discern the bangs (if $nbang > 1$, otherwise step 2).
2. Accurate positioning of single bang(s) (several if $nbang > 1$).

Program structure:

```
//Calculation 1:
//Discern the bangs if $nbang > 1$ else if $nbang = 1$ make accurate calculation

For (int i=0; i<ncomb; i++)
{
        SetStartValues(x0,y0,z0,t0);
        NewtonLoop(timecombination[ i ]);
}

//Calculation 2 (if $nbang > 1$)
//More accurate calculation for each bang separately

If($nbang > 1$)
For (int i=0; i<nbang; i++)
{
        SetStartValues(x,y,z,t of CorrectTimeComb[ i ]);
        NewtonLoop(timesforbang[ i ]);
}
```

Surveying a single bang (Newton Loop)

This calculation is the fundamental calculation in BANG. It is used for determining the near exact position and time point of one bang.

The cost function

For positioning a point in R^n, one must have $n + 1$ microphones (more theory on page 14). In the real 3D-world, at least four microphones are required to position one or more bangs. For simplicity's sake, a 2D system (figure **(1)**) with a minimum number of microphones is illustrated. This simple example is sufficient to understand the theory of BANG.

The engine of BANG minimises the so called cost function q **[1]**. The value of q (also known as the residual), is an indication of how close the approximated position and time are to the true position and time. It is the distance from the estimated position to the "true" position squared.

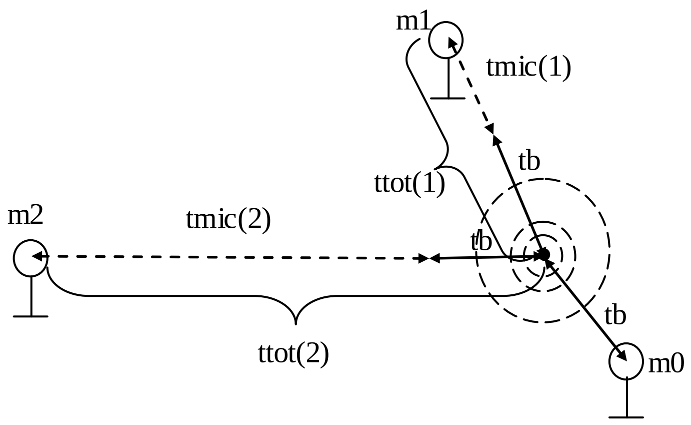
$$q = \sum_{n=1}^{nmic} \left(tmic(n) + tb - ttot(n)\right)^2 \qquad \textbf{[1]}$$

Where $tmic(n)$ is the difference between the smallest registered time and the time of registration at microphone n. tb is the current approximation of the time of the first registration and $ttot(n)$ is the distance from microphone n to the approximated position (xb, yb, zb). Note that $tmic(n)$ and tb are always scaled to distances with the reference speed of sound ($aspeed$). If weather distortion is taken into account, the total distance $ttot(n)$ will alter as the speed of sound will vary at different heights. See page 8 for details on calculating $ttot(n)$ with weather distortion. **[2]** shows $ttot(n)$ without weather taken into account.

$$ttot(n) = \sqrt{(xmic(n) - xb)^2 + (ymic(n) - yb)^2 + (zmic(n) - zb)^2} \qquad \textbf{[2]}$$

If the time and position are chosen correctly, $tmic(n) + tb - ttot(n)$ will be zero or close to zero from the bang to each microphone, so q will be minimal, ideally zero.

With four microphones spread out in the $x, y, z -$ space, any values of $tmic(n)$ for each microphone will lead to a fixed point. In other words q is bound to converge to exactly zero. With more than four microphones, the measurements of each $tmic(n)$ must be precisely accurate in order for $q = 0$. In the real world with imperfections and weather distortion q is not likely to reduce to exactly zero with more than four microphones. However, the more microphones used, the more accurate the result will be.



(1) *Shows the problem of BANG in* R^2 *with three microphones and one bang. When* xb, yb, zb *and* tb *are chosen correctly, the "puzzle pieces" will fit together and the correct position and time point of the bang is found.*

Minimising the cost function using Newton-Raphson

The cost function is minimised iteratively with a refined Newton-Raphson method. For a more thorough description, see **{1}**.

Start values

An initial guessed position and time point of the bang is required to start the convergence process. If an accurate guess cannot be made, there is an acceptable alternative. The guessed position's x and y coordinates $(x0, y0)$ are set to the average of the x and y coordinates of all microphones. The height $z0$ is just set to a height of 500 m. $t0$ is set to the smallest microphone registration time.

Calculating the next approximation

On each iteration, an improved estimate is calculated using the previous approximation, the first being the start values. This is done with a Newton-Raphson method with knowledge of the second derivative:

The value of the cost function q **[1]** at the current approximation point is calculated. The first and second derivatives of q with respect to x, y, z and t are then calculated by using the step length for numerical differentiation inc. See attachment 1 for details on the derivative calculations. The first and second derivatives of q:

$$q' = g = \begin{bmatrix} \frac{\delta q}{\delta x} & \frac{\delta q}{\delta y} & \frac{\delta q}{\delta z} & \frac{\delta q}{\delta t} \end{bmatrix} \qquad \textbf{[3]}$$

$$q'' = H = \begin{bmatrix} \frac{\delta^2 q}{\delta x^2} & \frac{\delta^2 q}{\delta x \delta y} & \frac{\delta^2 q}{\delta x \delta z} & \frac{\delta^2 q}{\delta x \delta t} \\ \frac{\delta^2 q}{\delta y \delta x} & \frac{\delta^2 q}{\delta y^2} & \frac{\delta^2 q}{\delta y \delta z} & \frac{\delta^2 q}{\delta y \delta t} \\ \frac{\delta^2 q}{\delta z \delta x} & \frac{\delta^2 q}{\delta z \delta y} & \frac{\delta^2 q}{\delta z^2} & \frac{\delta^2 q}{\delta z \delta t} \\ \frac{\delta^2 q}{\delta t \delta x} & \frac{\delta^2 q}{\delta t \delta y} & \frac{\delta^2 q}{\delta t \delta z} & \frac{\delta^2 q}{\delta t^2} \end{bmatrix} \qquad \textbf{[4]}$$

Where **[3]** is the gradient g with partial derivatives and **[4]** is the symmetrical hessian matrix H with the partial second derivatives.

By rewriting a truncated Taylor series **[5]**, the next approximation can be found by subtracting the Newton direction $g \cdot H^{-1}$ from the current approximation:

$$q_{k+1} = q_k - g_k \cdot H_k^{-1} \qquad \textbf{[5]}$$

The new estimate of $xb,\ yb,\ zb$ and tb is found with **[6]**.

$$\begin{aligned} xb_{k+1} &= xb_k - g_k(1) \cdot \left[H_k^{-1}(1{,}1) + H_k^{-1}(1{,}2) + H_k^{-1}(1{,}3) + H_k^{-1}(1{,}4)\right] \\ yb_{k+1} &= yb_k - g_k(2) \cdot \left[H_k^{-1}(2{,}1) + H_k^{-1}(2{,}2) + H_k^{-1}(2{,}3) + H_k^{-1}(2{,}4)\right] \\ zb_{k+1} &= zb_k - g_k(3) \cdot \left[H_k^{-1}(3{,}1) + H_k^{-1}(3{,}2) + H_k^{-1}(3{,}3) + H_k^{-1}(3{,}4)\right] \\ tb_{k+1} &= tb_k - g_k(4) \cdot \left[H_k^{-1}(4{,}1) + H_k^{-1}(4{,}2) + H_k^{-1}(4{,}3) + H_k^{-1}(4{,}4)\right] \\ &\text{Where } k = 0, 1, \ldots maxiter \end{aligned} \qquad \textbf{[6]}$$

Note that the numbers in brackets in **[6]** represent each index of the gradient and inverted hessian.

The inverted hessian is calculated by using Gauss elimination.

Comments

The reason for the use of second partial derivatives in the Newton-Raphson calculation is that convergence is reached with fewer iterations. The total calculation time will not necessarily be faster because of the calculation of the second derivatives and handling of the hessian matrix. In the case of larger calculations with many combinations (see page 11) and weather points (see the next section), the convergence may well be somewhat faster.

For an even faster convergence, the Newton direction can be multiplied by a scale factor λ, found with a backtracking line search method, see **{1}**. This is not yet implemented as it is not absolutely necessary for good results, more a finesse.

Perhaps the greatest downside of the Newton-Raphson is its poor global convergence properties. The prerequisite is that the start values are relatively close to the ”true” values. Convergence however, is still not a guarantee even if the start values are thought of as being qualified, though the chances of convergence will be high with accurate start values.

Calculating the speed of sound with weather distortion

As we have noted earlier, the times in the cost function are scaled to distances with the speed of sound. The speed of sound is not constant, but varies with the air properties and weather distortion.

The following weather data is measured with a weather balloon at certain heights going up over time:

$Pressure -$ air pressure
$Temp -$ temperature
$RH -$ relative humidity
$DewPTemp -$ Dewpoint temperature
$Dir -$ wind direction
$Speed -$ wind speed

Each sample is saved in a text file which BANG reads. Because of limited time for this work, BANG only takes wind and temperature into consideration. It is however, these two factors that typically have the most effect on the speed of sound and the two that vary the most (more in the next section).

Linear interpolation

As the weather data is specified at a number of discrete heights, linear interpolation is used for determining values at heights between the measured points. The number of new points calculated is determined by the step length dz. As we have discussed earlier, the measured and calculated times for each microphone are compared when calculating the residual q. The difference in height from the actual microphone to the bang is divided by the step length dz to get the number of points to be calculated. The points of height are evenly spread from the bang to the microphone. Points in between the measured points are calculated with linear interpolation. This determines the local speed of sound at each point. The distance from the bang to each microphone **[7]** $ttot$ is calculated by adding the step length along the true line from microphone to bang multiplied by the reference speed of sound divided by the local speed of sound to **[2]** for each interpolation point.

$$ttot = ttot + \sum_{i=1}^{nstep} \frac{aspeed}{ss(i)} \cdot ds \qquad \textbf{[7]}$$

Where the step length is **[8]**.

$$ds = s \cdot dz/(zb - zmic(n)) \quad \textbf{[8]}$$

Where s is the original $ttot$ from **[2]**.

Temperature calculations

Let's look at the Newton-Laplace equation for determining the speed of sound in a gas (air) **[9]**.

$$c = \sqrt{\gamma \cdot \frac{P}{\rho}} \quad \textbf{[9], \{2\}}$$

Where P is the air pressure, ρ is the air density and γ the adiabatic index of air (assumed $7/5 \approx 1{,}4$ for dry air **{2}**). We assume the air to be an ideal gas, thus the formula **[9]** can be rewritten as **[10]**.

$$c = \sqrt{\frac{\gamma \cdot R \cdot T}{M}} \quad \textbf{[10], \{2\}}$$

Where R is the gas constant, T is the absolute temperature in Kelvin and M the molar mass of air. The air pressure and air density are now not required. We have an expression with one dominating variable, the temperature T. The rest of the factors remain relatively constant. R, the molar gas constant, is always the same. M and γ vary slightly with the relative humidity of air.

For dry air $\gamma = 1{,}4$ and $M = 0{,}0289645\ kg \cdot mol^{-1}$, $R = 8{,}314510\ J \cdot mol^{-1} \cdot K^{-1}$.
Re writing **[10]** gives **[10']**.

$$c = 331{,}32 \cdot \sqrt{1 + \frac{\vartheta}{273{,}15}} \quad \textbf{[10'], \{2\}}$$

Where ϑ is the air temperature in °C.

The temperature could earlier be given as a constant, as ISA-profile or as a function of height in the old BANG. Now the weather data can only be given as a function of height since Bofors own weather system takes measurements as a function of height and writes them to a text file.

The temperature at a certain height h between measured points is calculated with linear interpolation **[11]**:

$$T(h) = T(h_0) + \frac{T(h_1) - T(h_0)}{h_1 - h_0} \cdot (h - h_0) \quad \textbf{[11], \{3\}}$$

Where h_0 and h_1 are the heights of the two weather points closest to h.

Wind calculations

The effect of the wind gradient on the speed of sound is even calculated (the x and y (rectangular form) components of the wind gradient as a function of height). When the effect of temperature has been calculated, the effect of wind is "added on" to the temperature influenced speed. The local x and y components of the wind are simply added on to the local speed of sound. The points between measured points are calculated in the same way as **[11]**, with linear interpolation. The x and y components are however calculated separately **[12]**:

$$W_x(h) = W_x(h_0) + \frac{W_x(h_1) - W_x(h_0)}{h_1 - h_0} \cdot (h - h_0)$$

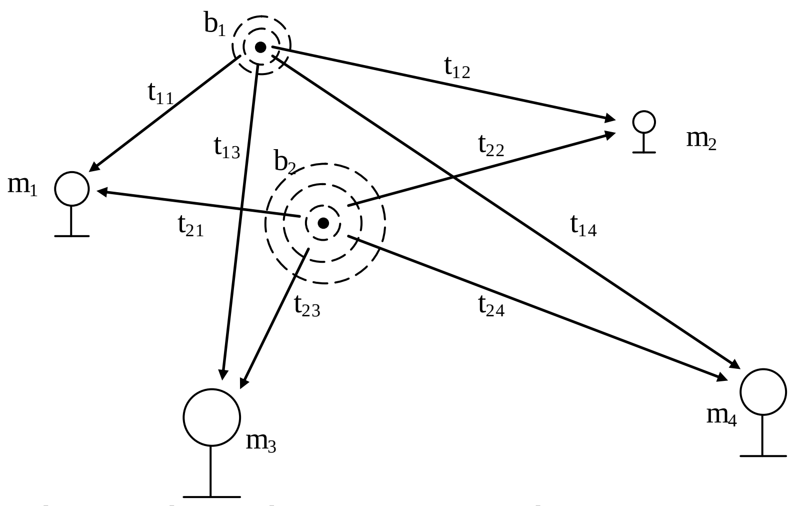
$$W_y(h) = W_y(h_0) + \frac{W_y(h_1) - W_y(h_0)}{h_1 - h_0} \cdot (h - h_0)$$

[12], {3}

Other about weather data

ISA stands for International Standard Atmosphere means the temperature vairies linearly with the height. The temperature is in fact often linear as a function of height up to around 11000 m altitude. Beyond this altitude the temperature remains roughly constant for a while: -56,6 °C **{4}**. The old BANG had the option of selecting this temperature type. This option is now however removed since weather data is always measured as a function of height and never assumed to vary linearly. The idea of this weather type should however not be forgotten as it may be desired in future use for time and data saving.

Discerning multiple bangs and sonic boom from bang



(2) *Shows a system with two bangs and four microphones in R^3. The first step is to determine which registration belongs to which bang at each microphone. A more accurate position and time point of each bang can thereafter be calculated according to the previous section.*

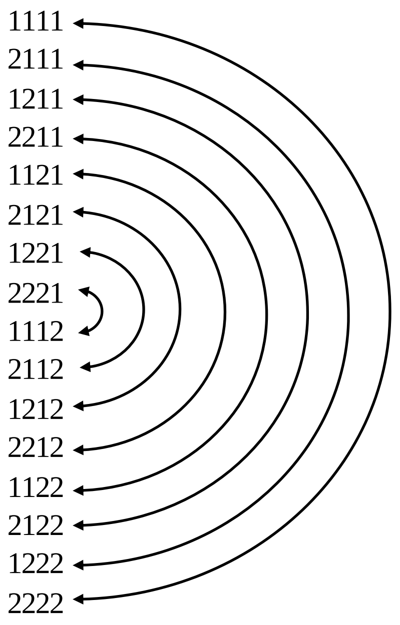
When more than one bang is present, more than one registration will be detected at each microphone. This is the case both for distinguishing several separations as well as a sonic boom from a separation. The question is: which of the registrations belong to which bang? This is initially unknown. See figure **(2)** with two bangs present.
Before an accurate position of each bang can be determined, the bangs must be discerned by finding the matching registrations (one at each microphone). This is done by calculating every combination separately with the refined Newton-Raphson method described in the previous section without the use of weather data. One registration at each microphone is tried at a time. The two matching combinations with the smallest summed residual q will most likely be the correct combinations. These two combinations are thereafter calculated individually as if they were two single bangs so their position can be calculated more precisely with weather data and adapted Newton Raphson data. Weather data is too time and data consuming, the speed of sound is therefore assumed to be constant everywhere for the discerning calculation.

The previous example is of simplest form with multiple bangs; We have two bangs and four microphones. The total number of combinations will be: $nbang^{nmic} = 2^4 = 16$. Each combination has a time array:

1111
2111
1211
2211
1121
2121
1221
2221
1112
2112
1212
2212
1122
2122
1222
2222

The number of digits in each time array corresponds to the number of microphones; In this case four. The number 1 represents the first registration and 2 the second at the microphone with the index of the digit's place. So "1111" means trying the combination of all the first microphone registrations, "1122" means trying the first registrations at microphone 1 and 2 and the second registrations at microphone 3 and 4.

The time arrays are matched together in the pairs that are logically possible **(3)**. The same registration cannot occur twice at the same microphone. The time arrays with opposite digits are therefore matched. So "1111" and "2222" is one possible outcome, "2121" and "1212" is another, etc.



(3) *The time arrays are matched with their compliments, the arrows indicating each matched pair.*

The residual for each pair is added **[13]**.

ResTot[1] = res [1111] + res [2222]
ResTot[2] = res [2111] + res [1222]
ResTot[3] = res [1211] + res [2122]
ResTot[4] = res [2211] + res [1122]
ResTot[5] = res [1121] + res [2212] **[13]**
ResTot[6] = res [2121] + res [1212]
ResTot[7] = res [1221] + res [2112]
ResTot[8] = res [2221] + res [1112]

The pair yielding the smallest residual is most likely the "true" combination order of times.

When the "true" combination is found, only a rough approximation of an acceptable result will have been obtained. The resulting position and time for each time array in the "correct" pair will be set as the start values for the separate (second) calculation, finding a more exact position and time for each bang.

Note that we can discern multiple bangs, but not determine in which order they occur. Because of long distances between microphones, the cargo grenades travel at the speed of sound or faster and that the second bang can occur much closer to a microphone than the first, means the second bang can be registered first. The order is determined with a video camera.

For the present, BANG can discern no more than two bangs. Surveying more than two bangs at a time is not presently necessary; however it may well be possible as described on page 24.

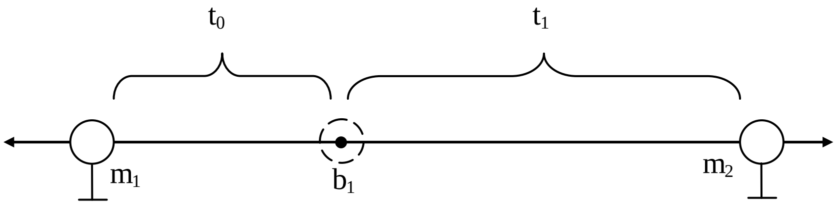
Geometrical analysis of the problem

In this section, the problem is analysed geometrically. A new graphical solution to the problem is discussed with an illustrative diagram from a new prototype Matlab program. This is an alternative solution with more or less guaranteed convergence. It however only works for one bang at the present and weather data is not taken into account, though it could easily be. The messy algebra, the inexact measurements and possibility of discerning multiple bangs are however strong arguments for the use of optimization as described earlier. The theory of the new method is explained in this section.

$n+1$ microphones are required for positioning in R^n. One of them takes the first registration and the others hold the time differences between the first registration and the time of registration at each microphone. In R^1 the time difference is represented as a line. In R^2 the time differences are represented as circles and in R^3 as spheres.
The other criterium for positioning in R^n is the microphones have to be spread out in R^n. So in R^1 the two microphones have to be placed on different points on the line. In R^2 and R^3 all microphones cannot be on the same line and for R^3 all microphones cannot be on the same plane. No microphones can offcourse be placed on the same point in all cases.

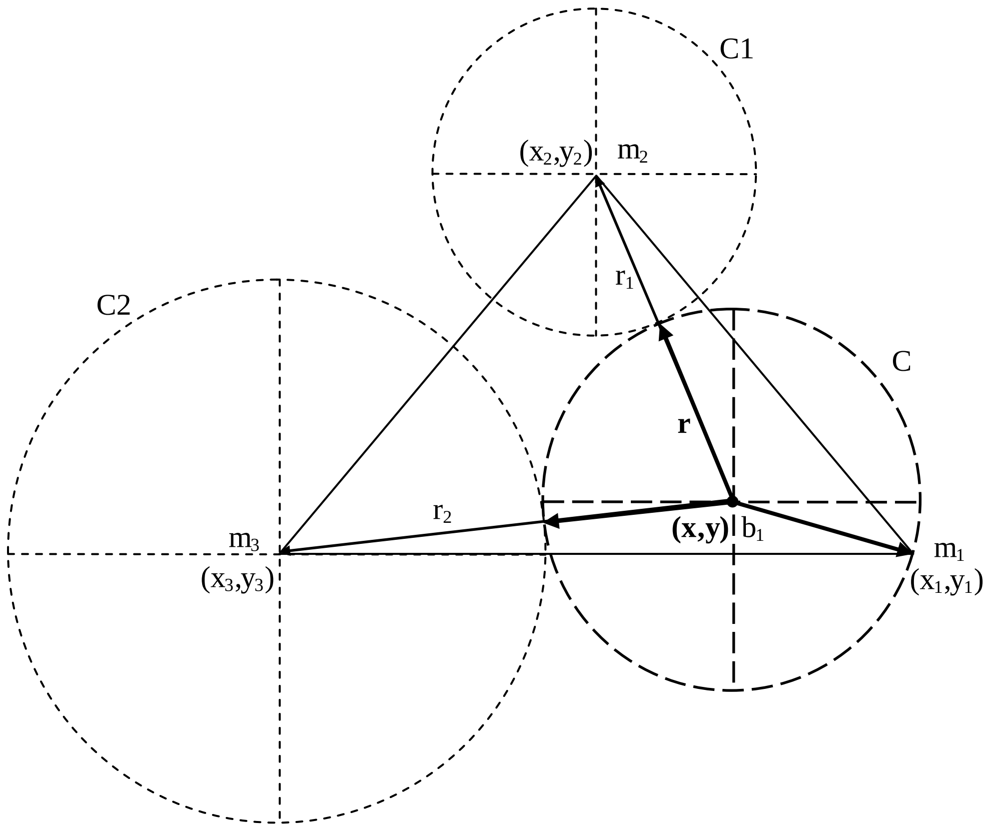
Surveying in one dimension

Let's start with imagining surveying a bang on a one dimensional line $\boldsymbol{R^1}$ **(4)**. The difference in registration time at microphone 1 (m_1) and microphone 2 (m_2) is easily determined. b_1*'s* distance from the middle point of the microphones is found by multiplying the difference in the times t_0 and t_1 with the speed of sound. b_1*'s* position will be this distance from the middle point of the microphones in the direction the microphone holding the smallest registration time, m_1.



(4) *The geometrical surveying system in simplest form on a line to illustrate the problem.*

Surveying in two dimensions



(5) *The 2D interpretation of the surveying system. The position and time point of b_1 correspond to the radius and centre point of the circle C.*

The same problem can be envisioned on a two-dimensional plane $\boldsymbol{R^2}$ **(5)**.

The circle C represents the propagation of the sound wave from b_1 at the time of the first microphone registration. The radius r of C is directly proportional to the time of the first registration, $tbtrue$. The radii r_1 and r_2 are directly proportional to $tmic(2)$ and $tmic(3)$, the time difference between the time of first registration and the time of registration at m_2 and m_3 respectively. The sought parameters are x, y and r.

Known and fixed parameters: $x_1, y_1, x_2, y_2, x_3, y_3, r_1, r_2$

Unknown and sought parameters: x, y, r

Three parameters are required to define a circle; the two coordinates for the centre point and the radius (alt. three of the rand points). Figure **(5)** can be described with the equation system **[14]**, consisting of three equations and three unknowns.

$$\begin{aligned}(x-x_1)^2+(y-y_1)^2&=(r+r_1)^2\\(x-x_2)^2+(y-y_2)^2&=(r+r_2)^2\\(x-x_3)^2+(y-y_3)^2&=r^2\end{aligned} \qquad \textbf{[14]}$$

An explicit solution to the system **[14]** exists, but the path to the solution is messy. Solving this system for two dimensions is a race with little prise, since the equivalent 3D-system still has to be solved. In the case of three dimensions, another term (z) and equation are added on. The thought of trying to solve this system **[15]** is horrid.

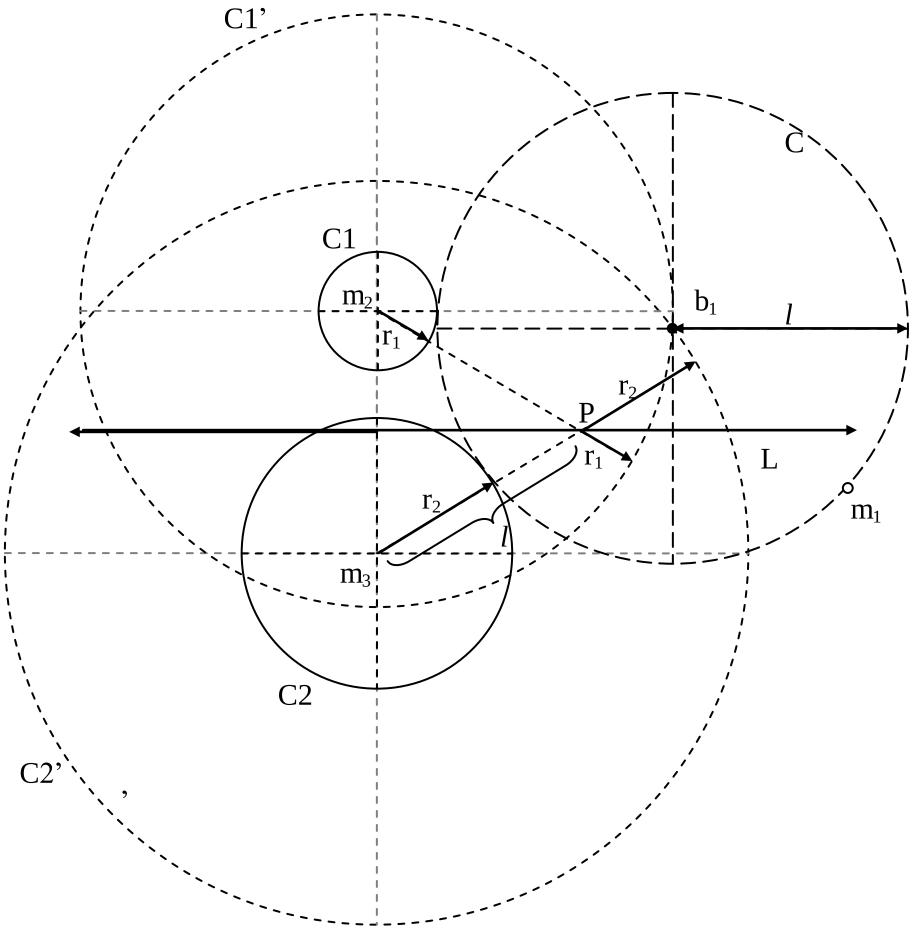
$$\begin{aligned}(x-x_1)^2+(y-y_1)^2+(z-z_1)^2&=(r+r_1)^2\\(x-x_2)^2+(y-y_2)^2+(z-z_2)^2&=(r+r_2)^2\\(x-x_3)^2+(y-y_3)^2+(z-z_3)^2&=(r+r_3)^2\\(x-x_4)^2+(y-y_4)^2+(z-z_4)^2&=r^2\end{aligned} \qquad \textbf{[15]}$$

Good luck in solving x, y, z and r...

A new version of BANG using a graphical solution

There is a graphical way around solving **[15]**. This is done with a prototype Matlab program, illustrated a little further on.

For simplicity, think back to the 2D system **(5)** with the circles. The middle point of a circle with any chosen radius l tangenting the two fixed circles $C1$ and $C2$ can be found. This is done by finding the intersection point of the circles $C1'$ and $C2'$ with their centre points in m_2 and m_3 and radii r_1+l and r_2+l. See figure **(6)**.



(6) *The principle of the new alternative graphical solution in R^2. The position and time point of b_1 is found by blowing up the circle C until it intersects the microphone of first registration m_1.*

The "bang-circle" C is a circle with a chosen radius l that tangents $C1$ and $C2$. This circle can be chosen to be of any size, as long as l is chosen so that both $C1'$ and $C2'$ intersect each other. The length l cannot however be determined so that C tangents m_1 at this stage.

Imagine instead a graphical program where the circle Ccan be blown up and shrunk down to what ever size providing it always tangents $C1$ and $C2$. The user changes l with a bar or textbox until C intersects the microphone of first registration m_1.

Trilateration

The same thing can be done in 3D-space with fixed spheres instead of circles. The "bang-sphere" S tangents three fixed spheres $S1$, $S2$ and $S3$ and is blown up manually by the user until it intersects the reference microphone. As with finding the intersection point of the two fixed circles $C1'$ and $C2'$, the intersection points of three spheres $S1'$, $S2'$ and $S3'$ are found with trilateration **[16]**, **{5}**.

$$X = \frac{r_1^2 - r_2^2 + x_2^2}{2x_2}$$
$$Y = \frac{r_1^2 - r_3^2 + x_3^2 + y_3^2}{2y_3} - \frac{x_3}{y_3} \cdot X \qquad \textbf{[16], \{5\}}$$
$$Z = \pm\sqrt{r_1^2 - X^2 - Y^2}$$

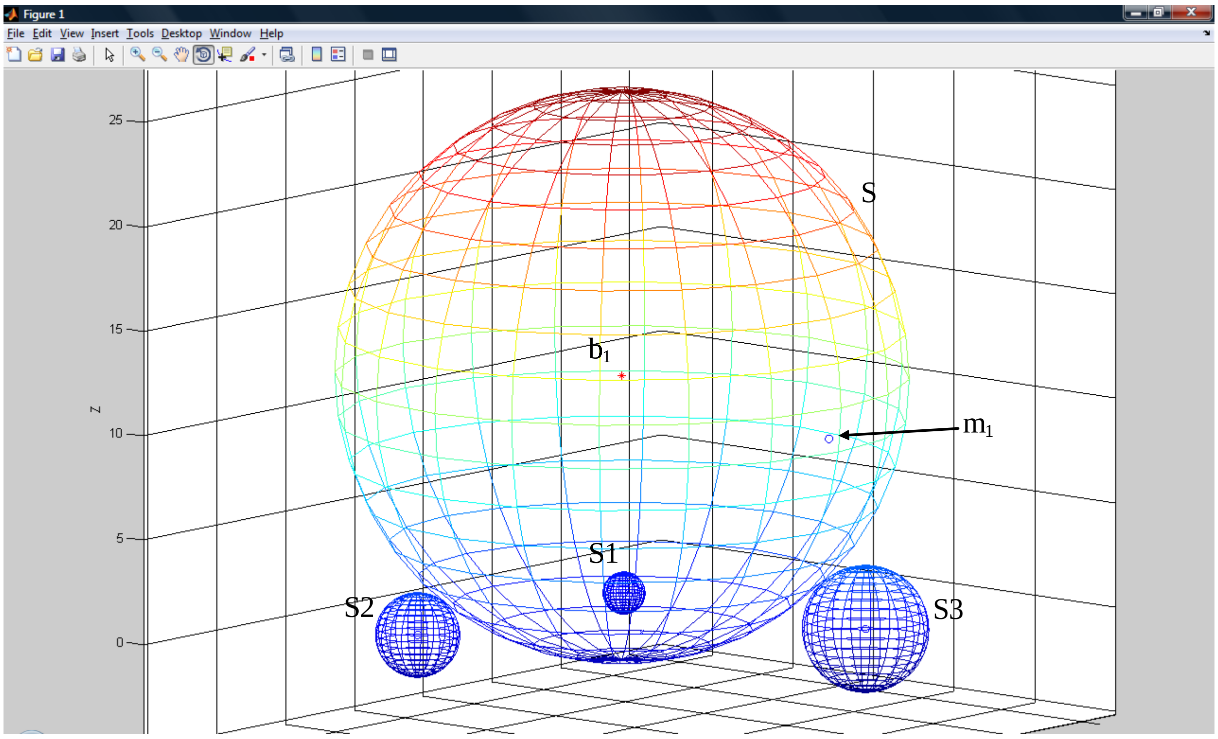
Where X, Y and Z are the coordinates for the two intersection points. Two solutions exist with two different values of Z. This is because the intersection of two spheres is a circle and a third sphere will intersect this circle in two points. Which of these points is the correct one can never be known with this method alone. A negative Z value can however be ruled out, since it is not likely the bang occurred below ground level. This is easily checked and the positive alternative is assumed the true one. One possibility is to use directed microphones, see page 25 or have a rough clue of where the bang is so the other alternative can be ruled out.

The Matlab program

Input data: microphone positions and the time data to the microphones. The radius of the "bang-sphere" can be fed in. The radius should be changeable by pulling a bar up and down on the graphics window.

Output data: graphical view of spheres and bang and microphone points. A dummy variable checking whether the reference microphone is inside the "bang-sphere" S.

With the input data, the program generates three fixed spheres: $S1$, $S2$ and $S3$ around the microphones m_2, m_3 and m_4 and the "bang-sphere" S which starts as its smallest possible size so its centre is positioned on the plane outlined by the points of m_2, m_3 and m_4. The user then increases its radius r until S tangents the reference microphone m_1. Each time r is changed, trilateration is used to calculate the new intersection points of $S1'$, $S2'$ and $S3'$. The new S is drawn out and a check on whether the reference microphone is inside S is made **(7)**.



(7) *Shows the position of the bang b_1 (red point surrounded by S) in relation to microphones (blue circles surrounded by $S1$, $S2$ and $S3$).*

This program is merely a prototype of an alternative solution and is not complete. Though an acceptably functional version would not require an exceeding amount of effort, this solution was not the focus of this work. It does however illustrate the problematic well and is thought to be used for quickly getting a rough result when only one bang. Another use of this prototype is finding more accurate start values from measured microphone data before running the original optimization program. The code of this program is much less and simpler than that of the original optimization program and a solution is very quickly reached. This makes the program an attractive alternative for a quick solution and easier to troubleshoot. Probably, the biggest plus with this solution is that convergence is almost a certainty since the spheres are "locked", a solution is always held with four microphones whatever the time differences should be. The Newton-Raphson method can however spin off to infinity should the start values not be sufficiently accurate. Newton-Raphson may even converge to the "wrong" solution as described on page 16. With the new graphical program, two solutions will be found with each system of four microphones. The "correct" solution is easily found if more than four microphones are used as one of the solutions in each system will be close to the others. The faulty Z value can be ruled out.

Note that such a sphere system can only be constructed with four microphones spread out in 3D-space. Real measurements will most probably be made with more than four microphones for more accurate results. In this case, three microphones are chosen at a time together with the reference microphone (with smallest registration time). Several of these systems can be made. For a more accurate positioning, the interesting result is the one given by the system of microphones with the smallest times. For an even more accurate result, the average of results of all systems can be made. The total number of systems that can be made **[17]**

$$nsystems = \frac{nmic}{4} \qquad \textbf{[17]}$$

is the number of microphones over four. Different methods using standard deviation can be used for a finer result than the average of all results.

Another alternative to changing r manually is by using a simple bisection method where the radius increases in equally sized steps until the reference microphone is inside S. The radius is then decreased with half the size of the original step. When the reference point is outside S again, the step size is halved again and radius increased until the tolerance is reached.

Note that this can only be done with systems of four microphones at a time. In addition, the microphones must be placed in an origin system with one microphone in origin, one on the $X-$ axis, one in the $XY-$ plane and one in $XYZ-$ space. Any set of four microphones can be converted to such a system, and thus converted back. This operation requires only a few simple matrix operations explained in the next section. The example in figure **(7)** is of simplest form, where m_2 is in origin, m_3 on the $X-$axes and m_4 on the $Y-$axes merely to illustrate that and how the system works.

Transition between coordinate systems in 3D-space

The following explains how a new coordinate system containing four microphones is constructed so that one microphone is placed in origin, one on the $X-$axes, one on the $XY-$ plane and the third in $XYZ-$ space. The condition is that the microphones are spread out in space in this way in the original system.

Let's start by defining a new system $=(X_S, Y_S, Z_S)$ in 3D-space.

A, B and C are the positions of three microphones in the original system.

1. Find the base vectors of the new system S.
   $V_1{}' = \overline{AB} = B - A$
   $V_2' = \overline{AC} - Proj_{V_1{}'}\overline{AC} = \overline{AC} - \frac{V_1{}' \cdot \overline{AC}}{V_1{}' \cdot V_1{}'} \cdot V_1{}'$
   $V_3' = V_1{}' \times V_2'$
   Where V_1', V_2' and V_3' are the base vectors for the new coordinate system S.

2. Normalise the base vectors:
   $V_1 = \frac{V_1{}'}{||V_1{}'||}, \quad V_2 = \frac{V_2{}'}{||V_2{}'||}, \quad V_3 = \frac{V_3{}'}{||V_3{}'||}$

3. Make a conversion matrix built up of the base vectors for the new system S:
   $$F = [V_1|V_2|V_3] = \begin{bmatrix} V_{11} & V_{21} & V_{31} \\ V_{12} & V_{22} & V_{32} \\ V_{13} & V_{23} & V_{33} \end{bmatrix}$$

   Also, calculate the inverse of F for the next step.

4. Calculate the coordinates of the microphones in the new system.
   We know that
   $[F] \cdot S = (Xold, Yold, Zold)$
   The coordinates of the new system will be:
   $S = (Xnew, Ynew, Znew) = [F^{-1}] \cdot (Xold, Yold, Zold)$

5. When the position of the bang has been found in the new system, convert back to the original system to get the true point:
   $(Xold, Yold, Zold) = [F] \cdot S = [F] \cdot (Xnew, Ynew, Znew)$

Procedure

Here is a short description of what was involved in the various steps of this work. A short list of the executed steps in order:

1. Analysis the old BANG Fortran code.
2. Structure diagrams for the new BANG, make skeleton and class diagram.
3. Make prototype code of the calculation modules and make sure the code can be compiled.
4. Make GUI so input data easily can be given for tests and output data read.
5. Test run the new BANG and compare results with old results.
6. When new results are identical to old, make fine adjustments to functionality and GUI.
7. Integrate Matlab for graphical presentation of result.
8. Document code and write a user manual.

The report writing began at step 6 and continued till the end of the work. Thoughts of other methods of improvement where thought of from the beginning of the work.

Discussion

How well does BANG work?

The study of how close the results of BANG are to the true results was not part of this work. This study would require much research and a whole new thesis. The goal of this work was to see that the new BANG gives identical results as the old BANG given identical input data. For the tested examples, this seems to be the case. A smaller test of how well the algorithm works could however be done without much difficulty by firing a gun from a measured position, so the position and time of the bang are known from the beginning.

Convergence

Unfortunately, BANG does not always converge because of Newton-Raphson’s poor global convergence properties. Qualified start values are required and even then convergence is not a guarantee. If the start values are sufficient however, convergence is fast. The graphical solution with spheres described on page 15 should be further examined to the fully, as a solution is guaranteed to be reached.

As described on page 16, we have the problem of two possible solutions. BANG can converge to the wrong solution. This must be checked somehow. The only check for the time being is if the height Z of the bang is below ground level. In this case, Z is simply set to its equivalent positive value. This may not be the correct value of Z so a few more iterations should be made before claiming the result reached. The new graphical algorithm will however converge to the correct solution if more than four microphones are used, see page 18.

Calculation time

The calculation time seems to vary greatly depending on the complexity of the system applied. Determining factors are the number of microphones, number of bangs, number of weather points and interpolation points of weather data (depending on the integration step length dz). The more of any of these factors applied, the longer the calculation time will be. In the case of two (or more in future) bangs, the time of the first half of the calculation is dependent on the number of microphones (and bangs) only. The time difference between using a few microphones and around 10 microphones is relatively large, but this calculation hardly takes one minute with 10 microphones and two bangs. Compared to the next calculation after the bangs have been separated, the first has little significance. The number of weather points and interpolation steps have the heaviest influence on the time, especially with a maximal number of bangs and microphones.

The effect of weather data

With a certain input data, the results of BANG with accounted weather data differ substantially from results without. What needs to be researched on is the reliability of the weather observations and weather calculations. Linear interpolation may give sufficiently accurate values of the points between measured points if the measured points are many and tightly together. With fewer data points with larger distance between them, it may be rewarding to use more advanced methods of interpolation with use of derivatives. This is more time and data consuming but the fewer weather points will hopefully even out the effort, in exchange for a more accurate result. Perhaps the degree of change in the measurements should be taken into account as the weather data will not remain constant.

The improvements made on BANG

C# VS Fortran

The program in C# is built up of classes in object oriented form. The classes make objects that are natural for the scenario; for instance: Microphone, Bang, Weather etc. An editor can quickly interpret the code and acquire an understanding of what's going should a new functionality be desired. This makes the new BANG is a good platform for new functionalities and improvements. The old Fortran code was substantially more difficult to edit as the entire code was stashed up in a single file. The C# version is faster and has a modern data structure.

Easy to use GUI

The new BANG has a user friendly GUI. Input data is easily fed in and changed. Tabs make it easy to flick between different input data such as microphone, bang and weather data. The result of BANG is easy to interpret. Positions and times of bangs can be viewed under the output tab. A graphical view of a compiled Matlab file can be viewed by pressing a button under the output tab.

Graphical view with Matlab

The result can be viewed graphically with an integrated Matlab program. The positions of microphones and bangs are viewed in a coordinate system that is rotatable, zoomable and possible to view in 2D with any of the two axes. These results are outputted from BANG to a text file read by the Matlab program when it is run. See details on how Matlab was integrated **{6}**.

Why Matlab?

Matlab is used because good looking and well functional 3D-graphics easily can be plotted. The Matlab graphics window has built in functions such as rotate, zoom and 2D-view. A good graphical solution is fully possible in C#. This would however demand much time in learning C# graphics. Functions such as rotation require extensive matrix operations. Problems arise with these matrix operations such as gimbal lock **{7}**. Gimbal lock is however avoidable with a quaternion **{8}** solution. A nice graphical solution in C# might be preferred in the long run as the complied Matlab program takes a while to start up. Less debug files would then be required and Matlab text files would not have to be created.

All calculations for multiple bangs are executed automatically

The old BANG had to be run several times in the case of more than one bang. The first execution was to discern the bangs. One extra execution was required for each bang to calculate a more accurate position for each bang once the correct combination order of time registrations is known. Weather data was taken into account on these second calculations. This meant having to feed in the results of the first calculation as start values for the second calculations. The new BANG makes all these calculations automatically. Weather data is given from the beginning and the program only has to be executed once.

Possibilities of further improvement

Here are some ideas of possible improvement.

Structs VS classes in C#

For larger, more time demanding calculations, it may be worth implementing structs instead of classes. This is especially worthwhile for the class handling the cost function “Cost”.

Improvements in weather calculations

The wind gradient should be able to be given in cardinal coordinates. These coordinates are converted to the reference coordinate system used. This is simply done by knowing the phase difference between the cardinal and reference coordinate systems.

Relative humidity in the weather calculations

The speed of sound increases with increased humidity. The relative humidity influences two variables in equation **[10]**, namely γ and M. The ratio between the speed of sound in dry air and in wet air will be **[18]**.

$$\sqrt{\frac{\gamma_w/M_w}{\gamma_d/M_d}} \qquad \textbf{[18], \{9\}}$$

Where γ_w and M_w are γ and M in ”wet” air and γ_d and M_d are γ and M in dry air. The “wet” variables can be calculated **[19]**.

$$\gamma_w = \frac{7+h}{5+h}$$
$$M_w = M_d \qquad \textbf{[19], \{9\}}$$

Where h is the fraction of water molecules in the air. h can be determined with **[20]**.

$$h = \frac{0{,}01 \cdot RH \cdot e^{20{,}386-5132/T}}{P} \qquad \textbf{[20], \{9\}}$$

Where P is the air pressure.

The humidity has a smaller effect on the speed of sound. The question is how much more accurate would the results from BANG be if RH was taken into account? Is the influence of RH greater than the deviation in weather observations and results from BANG? Research is needed to answer this question.

Frequency analysis for discerning sonic boom

The possibility of using frequency analysis for discerning the sonic boom should be examined. This will most likely be more efficient than the combination method described on page 11.

Is the dummy variable time_known necessary?

The question arose if the possibility of setting the time of first registration to known is necessary. This means that the guess time remains unchanged throughout the entire calculation and is not updated with a new approximation. Knowing the time means minimising the calculation time substantially, by roughly a quarter. This is more noticeable in the case of larger calculations. In a test scenario, measuring the flight time can be worth the effort, and having the option of changing “time_known” does not hurt.

Discerning more than two bangs

If the need of positioning more than two bangs should arise, an expansion of the current method used is not that difficulty implemented. For details on how this method works, se page 11.

Earlier we have seen that each possible combination is given a time array. This is the case even here with more than two bangs. An example is illustrated below with three bangs and three microphones:

111
211
311
121
221
321
131
231
331
112
212
312
122
222
322
132
232
332
113
213
313
123
223
323
133
233
333

Naturally, at least four microphones must be present for measurements in 3D-space. This simple example is merely to show the principle and does not need to be more complicated to understand the idea.

As in the case of two bangs, all combinations are matched with their opposite compliment. The difference is that more than one combination will match each combination. Examples of matches:

111 matches with the following: 222, 322, 232, 332, 223, 323, 233 and 333.

If 111 and 222 $\longrightarrow$ 333
If 111 and 322 $\longrightarrow$ 233
etc.

The next combination 211 matches the following: 122, 322, 132, 332, 123, 323, 133 and 333.

If 211 and 122 $\longrightarrow$ 333
If 211 and 322 $\longrightarrow$ 133
etc.

As for two bangs, the residuals are added for each possible set of matched combinations:

Restot1 = res [111] + res [222] + res [333]
Restot2 = res [111] + res [322] + res [233]
etc…

The method can theoretically be used for an unlimited number of bangs.

This method may not be actual for larger calculations and more than two bangs. The amount of time that can be spent is the deciding factor of using this method. BANG is however not designed for more than two bangs, since it is not used for more than two. For more, other methods should be examined.

Choice of microphones

Condenser microphones should be used as the sound from the bangs is very short and sharp, having a chaotic nature. The capacitor effect will capture the steep slopes of the impulse from the bang.
The microphones should even be of a directed type, not an isotropic microphone so there is a possibility of ruling out a faulty convergence see page 16.

References

{1} http://trond.hjorteland.com/thesis/node28.html 11/07/01

{2} http://en.wikipedia.org/wiki/Speed_of_sound 11/07/01

{3} http://translate.google.se/translate?hl=sv&langpair=en|sv&u=http://en.wikipedia.org/wiki/Linear_interpolation 11/07/01

{4} http://s6.aeromech.usyd.edu.au/aero/atmosphere/atmosphere.pdf 11/07/01

{5} http://en.wikipedia.org/wiki/Trilateration 11/07/01

{6} http://pages.stern.nyu.edu/~nwhite/scrc/CompilingMatlab.htm for the integration of Matlab in C# 11/07/01

{7} http://en.wikipedia.org/wiki/Gimbal_lock 11/07/01

{8} http://en.wikipedia.org/wiki/Quaternion 11/07/01

{9} http://www.rane.com/pdf/ranenotes/Enviromental%20Effects%20on%20the%20Speed%20of%20Sound.pdf 11/07/01

{10} programbeskrivning BANG, separationspunktsbestämning mha ljud / SIMBAL 11/07/01

Appendix

Appendix 1 – Derivative calculations

The gradient *g*:

$$\frac{\delta q}{\delta x}=\frac{q(xb+inc,yb,zb,tb)-q(xb,yb,zb,tb)}{inc}$$
$$\frac{\delta q}{\delta y}=\frac{q(xb,yb+inc,zb,tb)-q(xb,yb,zb,tb)}{inc}$$
$$\frac{\delta q}{\delta z}=\frac{q(xb,yb,zb+inc,tb)-q(xb,yb,zb,tb)}{inc}$$
$$\frac{\delta q}{\delta t}=\frac{q(xb,yb,zb,tb+inc)-q(xb,yb,zb,tb)}{inc}$$

The hessian *H*:

$$\frac{\delta^2 q}{\delta x^2}=\frac{q(xb+inc,yb,zb,tb)-q(xb,yb,zb,tb)-q(xb,yb,zb,tb)+q(xb-inc,yb,zb,tb)}{inc^2}$$
$$\frac{\delta^2 q}{\delta x\delta y}=\frac{q(xb+inc,yb+inc,zb,tb)-q(xb+inc,yb,zb,tb)-q(xb,yb+inc,zb,tb)+q(xb,yb,zb,tb)}{inc^2}$$
$$\frac{\delta^2 q}{\delta x\delta z}=\frac{q(xb+inc,yb,zb+inc,tb)-q(xb+inc,yb,zb,tb)-q(xb,yb,zb+inc,tb)+q(xb,yb,zb,tb)}{inc^2}$$
$$\frac{\delta^2 q}{\delta x\delta t}=\frac{q(xb+inc,yb,zb,tb+inc)-q(xb+inc,yb,zb,tb)-q(xb,yb,zb,tb+inc)+q(xb,yb,zb,tb)}{inc^2}$$

$$\frac{\delta^2 q}{\delta y\delta x}=\frac{q(xb+inc,yb+inc,zb,tb)-q(xb+inc,yb,zb,tb)-q(xb,yb+inc,zb,tb)+q(xb,yb,zb,tb)}{inc^2}$$
$$\frac{\delta^2 q}{\delta y^2}=\frac{q(xb,yb+inc,zb,tb)-q(xb,yb,zb,tb)-q(xb,yb,zb,tb)+q(xb,yb-inc,zb,tb)}{inc^2}$$
$$\frac{\delta^2 q}{\delta y\delta z}=\frac{q(xb,yb+inc,zb+inc,tb)-q(xb,yb+inc,zb,tb)-q(xb,yb,zb+inc,tb)+q(xb,yb,zb,tb)}{inc^2}$$
$$\frac{\delta^2 q}{\delta y\delta t}=\frac{q(xb,yb+inc,zb,tb+inc)-q(xb,yb+inc,zb,tb)-q(xb,yb,zb,tb+inc)+q(xb,yb,zb,tb)}{inc^2}$$

$$\frac{\delta^2 q}{\delta z\delta x}=\frac{q(xb+inc,yb,zb+inc,tb)-q(xb+inc,yb,zb,tb)-q(xb,yb,zb+inc,tb)+q(xb,yb,zb,tb)}{inc^2}$$
$$\frac{\delta^2 q}{\delta z\delta y}=\frac{q(xb,yb+inc,zb+inc,tb)-q(xb,yb+inc,zb,tb)-q(xb,yb,zb+inc,tb)+q(xb,yb,zb,tb)}{inc^2}$$
$$\frac{\delta^2 q}{\delta z^2}=\frac{q(xb,yb,zb+inc,tb)-q(xb,yb,zb,tb)-q(xb,yb,zb,tb)+q(xb,yb,zb-inc,tb)}{inc^2}$$
$$\frac{\delta^2 q}{\delta z\delta t}=\frac{q(xb,yb,zb+inc,tb+inc)-q(xb,yb,zb+inc,tb)-q(xb,yb,zb,tb+inc)+q(xb,yb,zb,tb)}{inc^2}$$

$$\frac{\delta^2 q}{\delta t\delta x}=\frac{q(xb+inc,yb,zb,tb+inc)-q(xb+inc,yb,zb,tb)-q(xb,yb,zb,tb+inc)+q(xb,yb,zb,tb)}{inc^2}$$

$$\frac{\delta^2 q}{\delta t \delta y} = \frac{q(xb, yb + inc, zb, tb + inc) - q(xb, yb + inc, zb, tb) - q(xb, yb, zb, tb + inc) + q(xb, yb, zb, tb)}{inc^2}$$

$$\frac{\delta^2 q}{\delta t \delta z} = \frac{q(xb, yb, zb + inc, tb + inc) - q(xb, yb, zb + inc, tb) - q(xb, yb, zb, tb + inc) + q(xb, yb, zb, tb)}{inc^2}$$

$$\frac{\delta^2 q}{\delta t^2} = \frac{q(xb, yb, zb, tb + inc) - q(xb, yb, zb, tb) - q(xb, yb, zb, tb) + q(xb, yb, zb, tb - inc)}{inc^2}$$

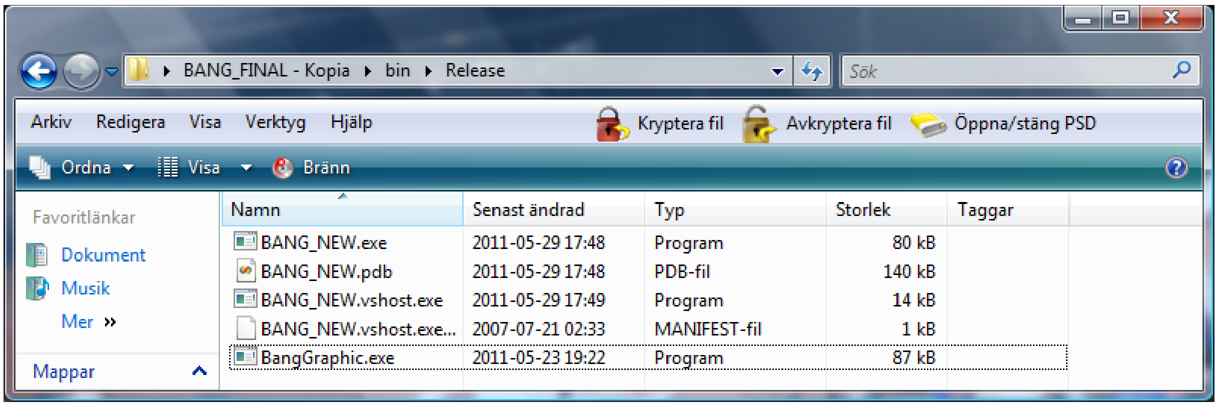
Appendix 2 – User manual

Installation

A list of all necessary complied and debug files for running BANG is shown in figure **(1)**.

To run the program, double click on BANG_NEW.exe.

A doxy file is also provided for detailed documentation of the code. This is not showing here, but is included in the program catalogue **(1)** and can be opened with the doxy wizard Doxygen.



Namn	Senast ändrad	Typ	Storlek	Taggar
BANG_NEW.exe	2011-05-29 17:48	Program	80 kB	
BANG_NEW.pdb	2011-05-29 17:48	PDB-fil	140 kB	
BANG_NEW.vshost.exe	2011-05-29 17:49	Program	14 kB	
BANG_NEW.vshost.exe...	2007-07-21 02:33	MANIFEST-fil	1 kB	
BangGraphic.exe	2011-05-23 19:22	Program	87 kB	

(1) *Shows the project catalogue with program and debug files. The output data files will be saved in this catalogue automatically while the input data files are saved in a chosen catalogue .*

Input and output data

Input data:

Microphone data:

- $nmic -$ number of microphones (maximum 10 for the present)
- $xmic(n), ymic(n), zmic(n) -$ the position of microphone n
- $tmic(n) -$ registration time at microphone n

Bang data:

- $nbang -$ number of bangs (maximum two for the present)
- $x0, y0, z0, t0 -$ Start values for position and time of one bang (first registered bang)
- $time\_known -$ If the time point for the first registration is known (true/false)

Weather data:

- $time -$ time of measurement (s)
- $Z -$ height of measurement (m)
- $Pressure -$ air pressure at weather point (hPa)
- $Temp -$ temperature at weather point $(°C)$
- $RH -$ relative humidity at weather point $(\%)$
- $DewPDeg -$ the decrease in temperature at weather point required for water vapour to condensate to water $(°C)$
- $Dir -$ direction of wind gradient at weather point $(°)$
- $Speed -$ speed of the wind gradient at weather point (m/s)

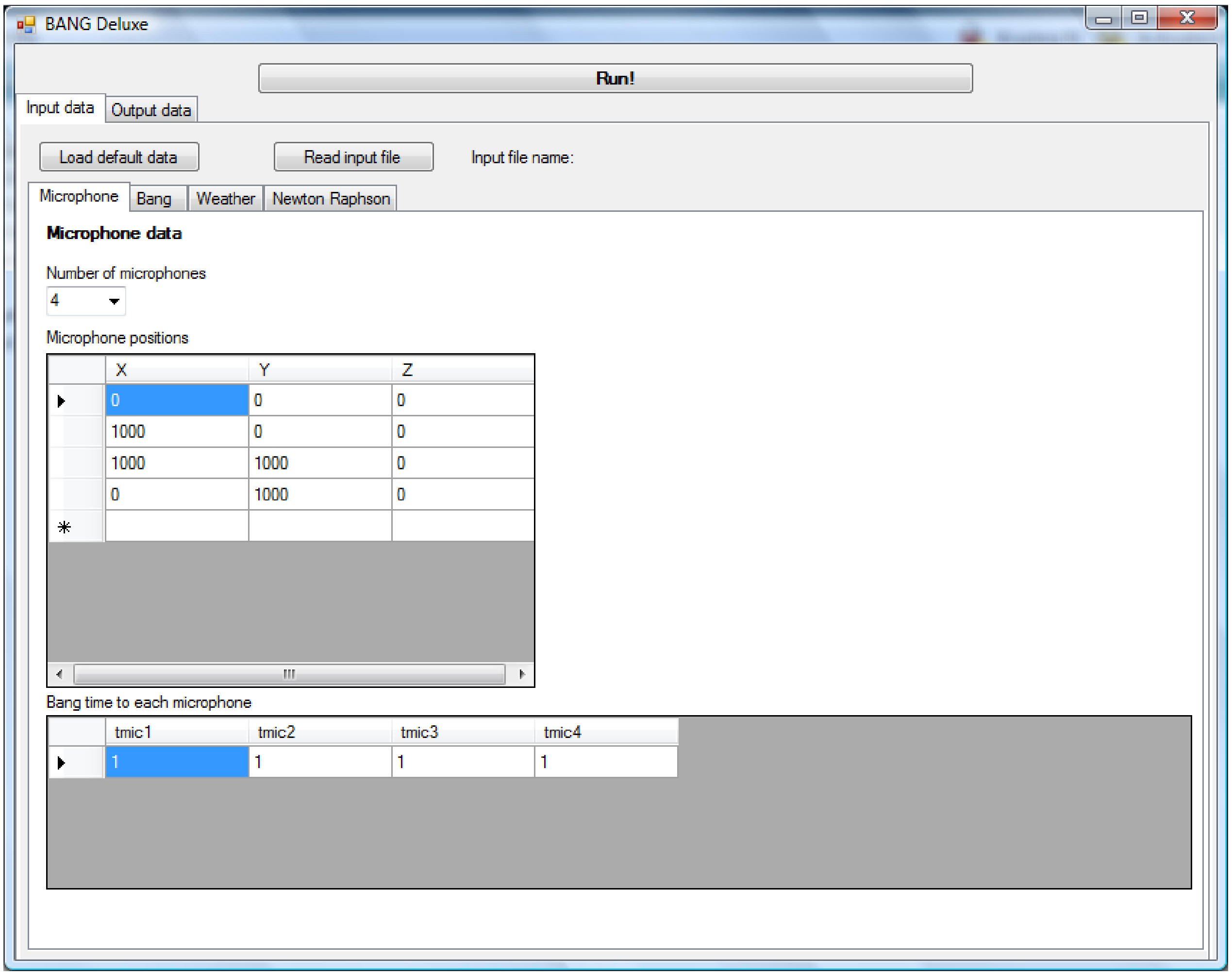
Newton-Raphson data:

- $maxiter -$ maximal number of iterations
- $resmin -$ largest residual for convergence
- $resmax -$ smallest residual for divergence
- $dz -$ integration step in height for interpolation of weather points
- $inc -$ step length for numerical derivation
- $sf -$ scale factor

Output data:

- $X\_true, Y\_true, Z\_true, T\_true -$ Position and time point of bang(s)
- Input file for Matlab with bang and microphone positions
- Graphical view of microphone and bang positions with Matlab graphics window

GUI



Microphone data

Number of microphones

4

Microphone positions

X	Y	Z
0	0	0
1000	0	0
1000	1000	0
0	1000	0

Bang time to each microphone

tmic1	tmic2	tmic3	tmic4
1	1	1	1

(2) *Tab for inputting microphone data*

When running the program, the GUI will appear **(2)**.

"Run!" – Press to run the calculation with the currently selected input data. A "browse window will pop up. Choose the path where you want to save the input file.

Input data tab – Here, the input data is applied.

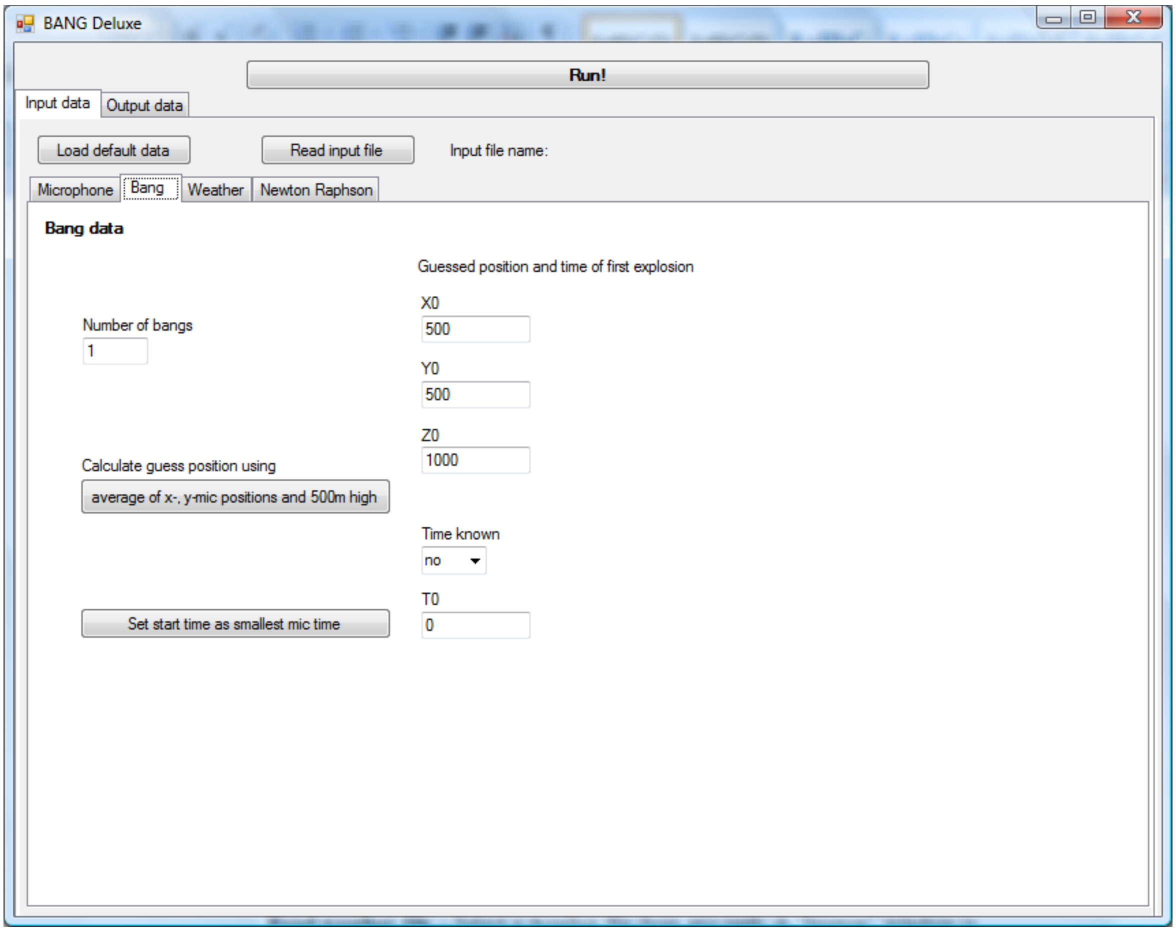
Load default data – Resets to the default input data so BANG can be tested.

Read input file – For selecting an input data file. A "browse" window pops up and an input data file can be selected from any path.

Under the input data tab, there are several sub tabs:

Microphone data: Select the number of microphones, the microphones positions and the registration times for each microphone **(2)**.

Bang data:



Bang data

Guessed position and time of first explosion

Number of bangs: 1

X0: 500

Y0: 500

Z0: 1000

Calculate guess position using: average of x-, y-mic positions and 500m high

Time known: no

T0: 0

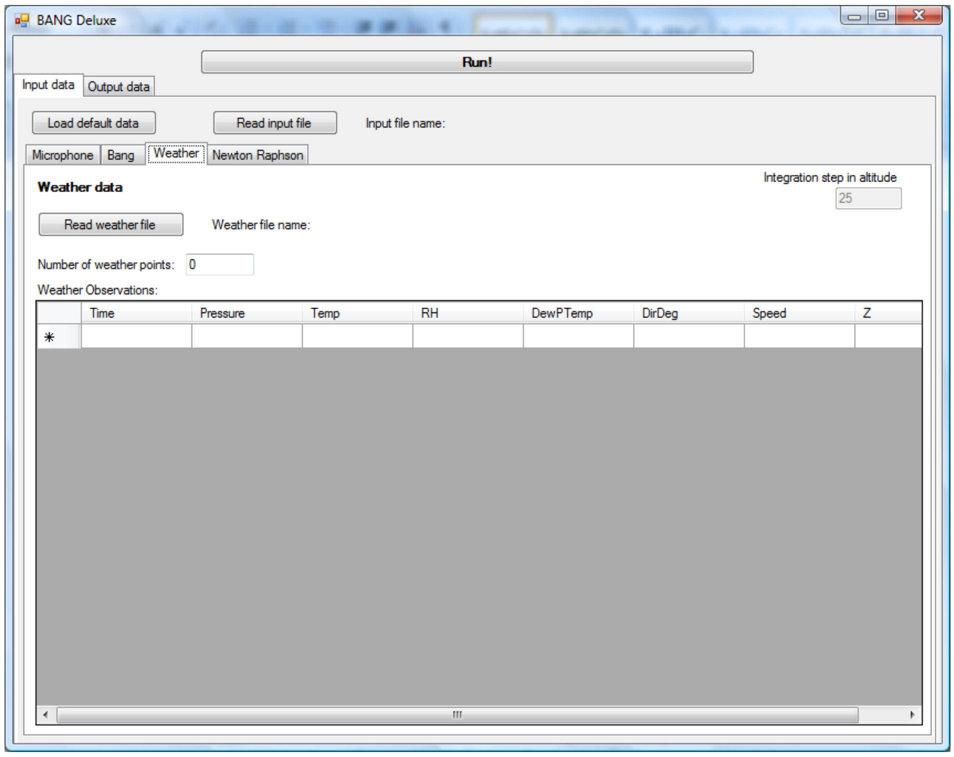
Set start time as smallest mic time

(3) *Tab for inputting bang data*

Select the number of bangs, start values and if the first registration time is known or unknown (default) **(3)**.

Weather data:
Read weather file – Select a weather file from any path. A "browse" window is opened.

The weather data is observed from a weather balloon always given as a function of height. Included measurements from each weather point: $time, Z, Pressure, Temp, RH, DewPTemp, Dir$ and $Speed$. Weather data is edited in window **(4)**.



Input file name:

Weather data

Integration step in altitude: 25

Weather file name:

Number of weather points: 0

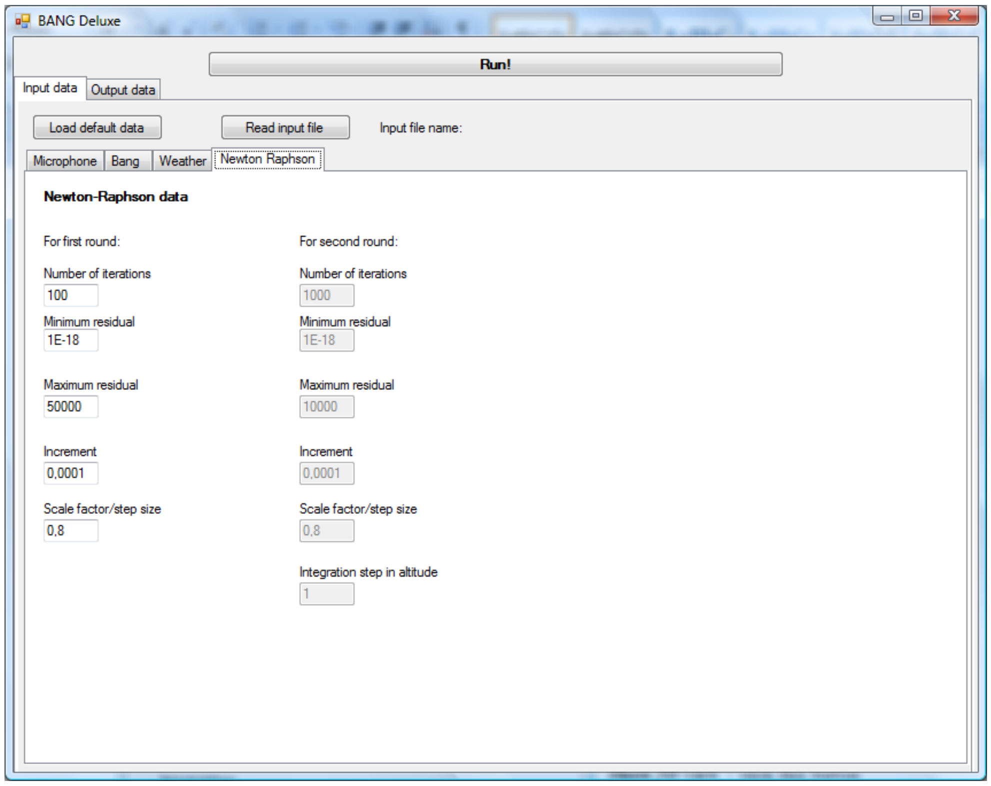
Weather Observations:

	Time	Pressure	Temp	RH	DewPTemp	DirDeg	Speed	Z
*								

(4) *Tab for inputting weather data*

Newton-Raphson :

Set $maxiter, resmin, resmax, inc$ and sf. Set these parameters for the second calculation when $nbang > 1$. If $nbang > 1$, set also dz for the second calculations **(5)**.



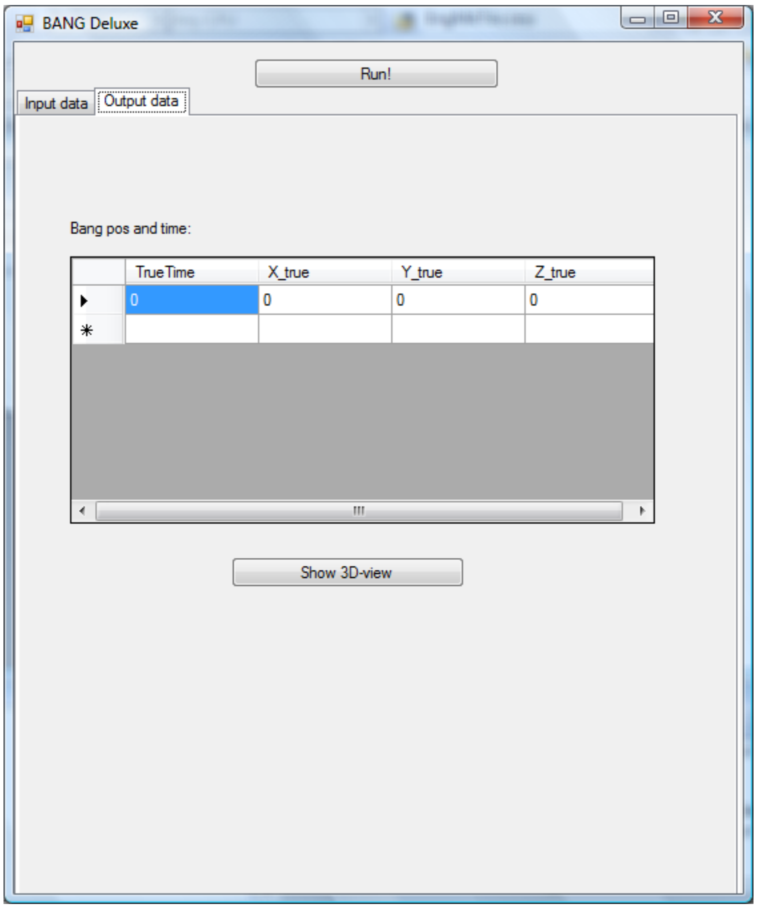
Input file name:

Newton-Raphson data

For first round:		For second round:	
Number of iterations	100	Number of iterations	1000
Minimum residual	1E-18	Minimum residual	1E-18
Maximum residual	50000	Maximum residual	10000
Increment	0,0001	Increment	0,0001
Scale factor/step size	0,8	Scale factor/step size	0,8
		Integration step in altitude	1

(5) *Tab for inputting Newton-Raphson data*

Output data tab:
The results of BANG are shown in the datagridview in the output tab **(6)**.



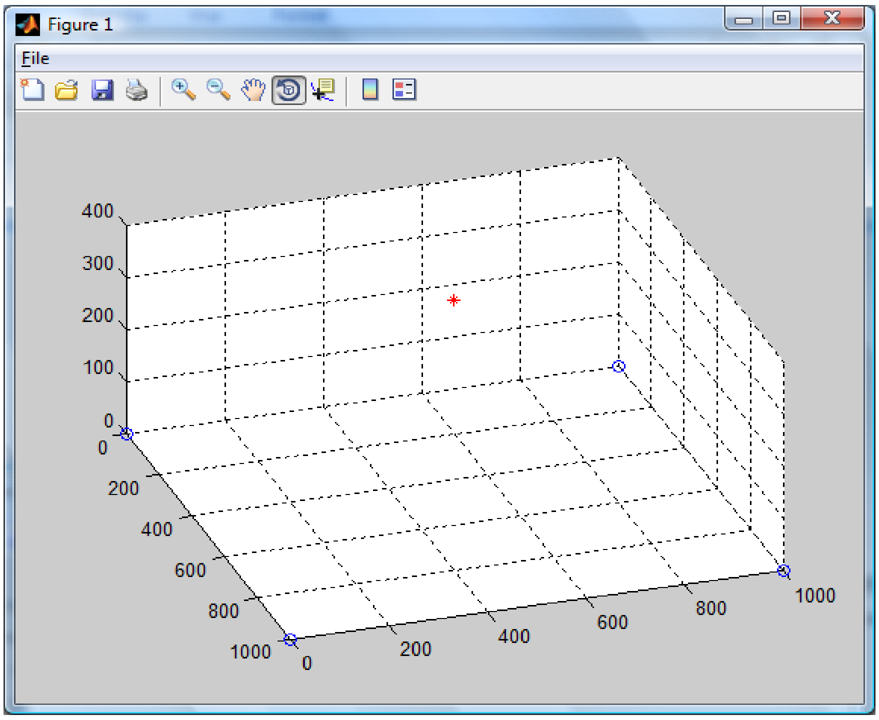
Bang pos and time:

	TrueTime	X_true	Y_true	Z_true
▸	0	0	0	0
*				

Show 3D-view – click this button for 3D-viewing of result. A Matlab graphics window with graphical view will appear **(7)** after a minute. This can be rotated with the built in ”Rotate 3D” button by dragging the image with the mouse. It’s also possible to zoom in and out and set 2D-views.

(6) *Tab for viewing output data. The position and time point of bang(s) are shown in the datagridview.*

The graphical result from running the default input data **(7)**.

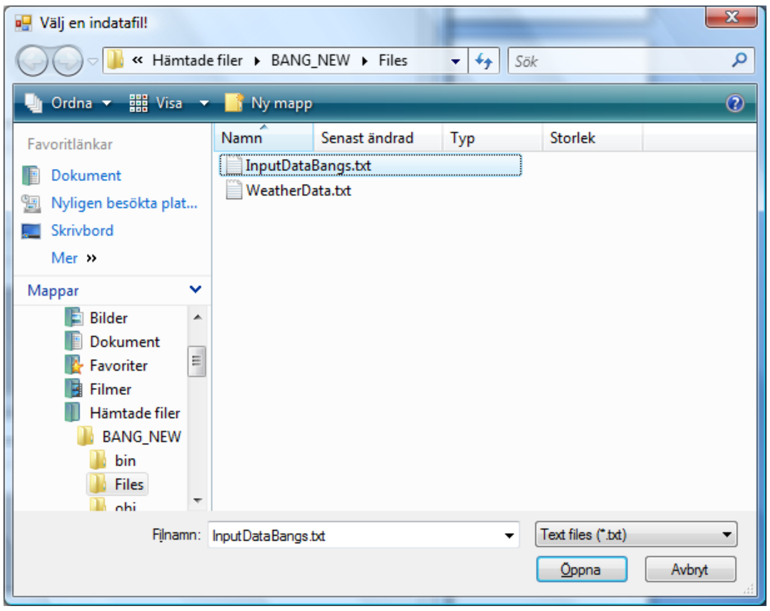


(7) *Graphical result of BANG. The blue circles represent the microphones and the red stars the bangs.*

Appendix 3 - Example of a test run

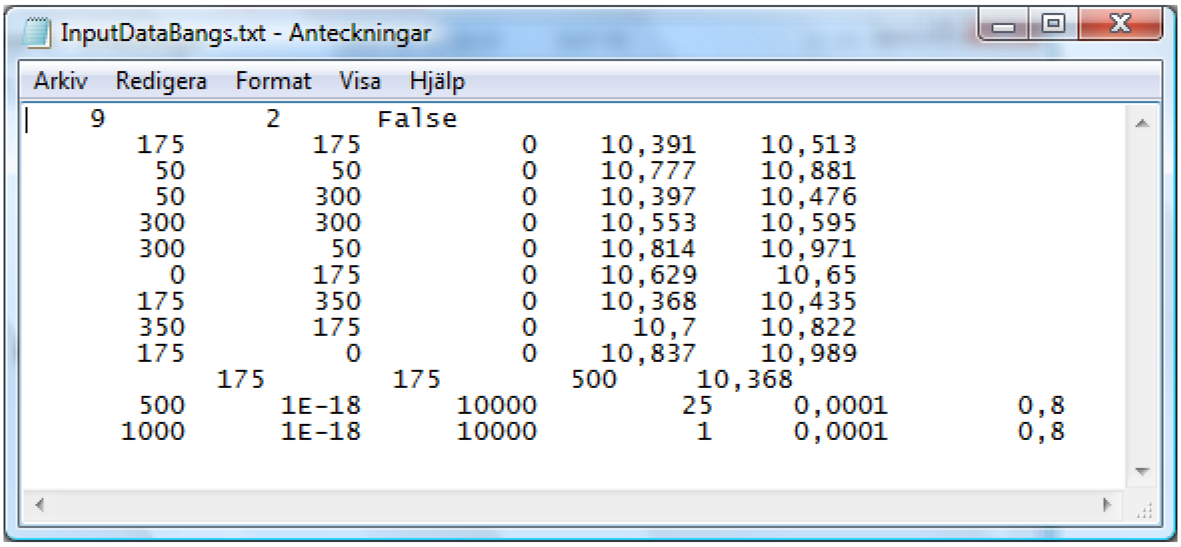
An example of running BANG. This is the same test data as given in **{10}**. The new BANG gives the same results as the old BANG for this input data.

Press ”**Read input file**”, window **(1)** will appear.



(1) *Select an input file here.*

Choose and input data file. Here the file is called ”InputDataBangs.txt” (we have two bangs). **(2)** shows the input data file opened.



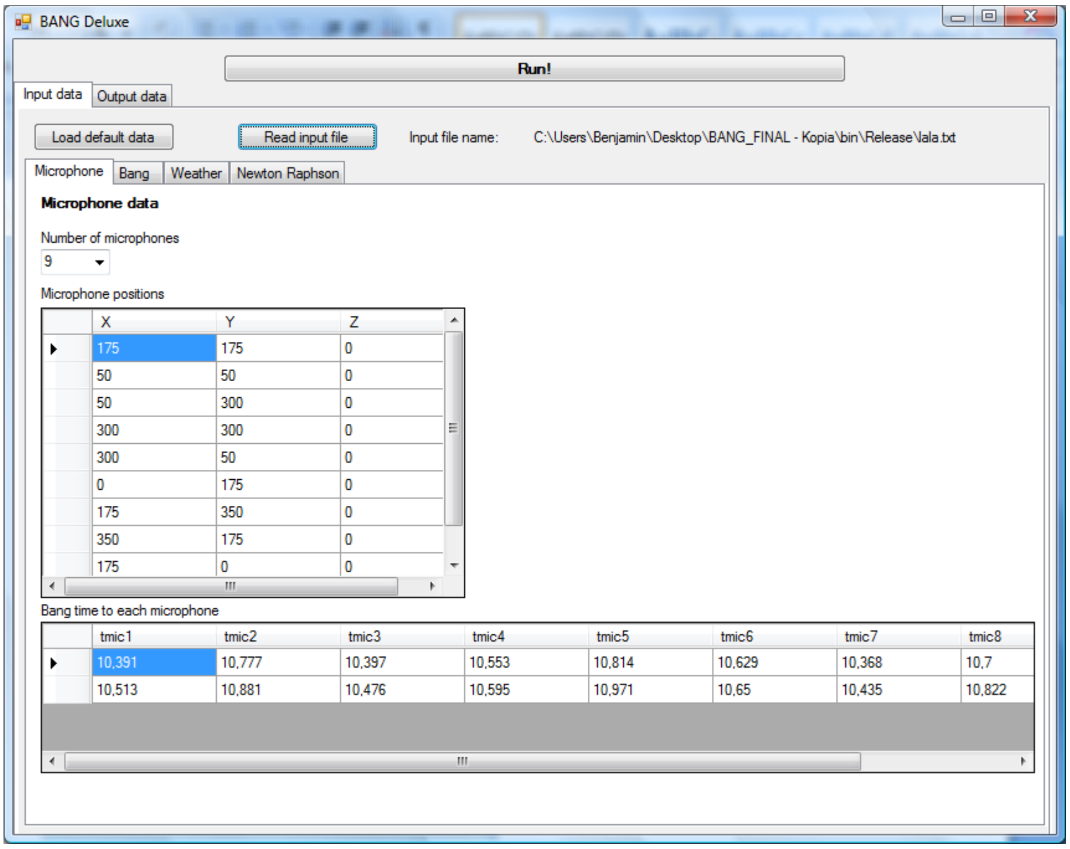
```
      9           2       False
          175         175           0     10,391     10,513
           50          50           0     10,777     10,881
           50         300           0     10,397     10,476
          300         300           0     10,553     10,595
          300          50           0     10,814     10,971
            0         175           0     10,629      10,65
          175         350           0     10,368     10,435
          350         175           0       10,7     10,822
          175           0           0     10,837     10,989
                175         175         500     10,368
          500       1E-18       10000          25      0,0001        0,8
         1000       1E-18       10000           1      0,0001        0,8
```

(2) *Input data file (.txt).*

We have 9 microphones, 2 bangs, the first registration time is unknown, the 9 microphone positions, time 1 and 2 for each microphone, the start guess $x0 = 175, y0 = 175, z0 = 500$ and $t0 = 10{,}368$. Because $nbang > 1$ we have two rows of Newton-Raphson parameters, one for the first calculation discerning the bangs and one for the second more accurate calculations for each bang.

When selecting this file, we can see that the tabs ”microphone data” **(3)**, ”bang data” **(4)** and “Newton-Raphson” **(5)** show the data from the selected file.

Microphone data



Input file name: C:\Users\Benjamin\Desktop\BANG_FINAL - Kopia\bin\Release\lala.txt

Microphone data

Number of microphones: 9

Microphone positions

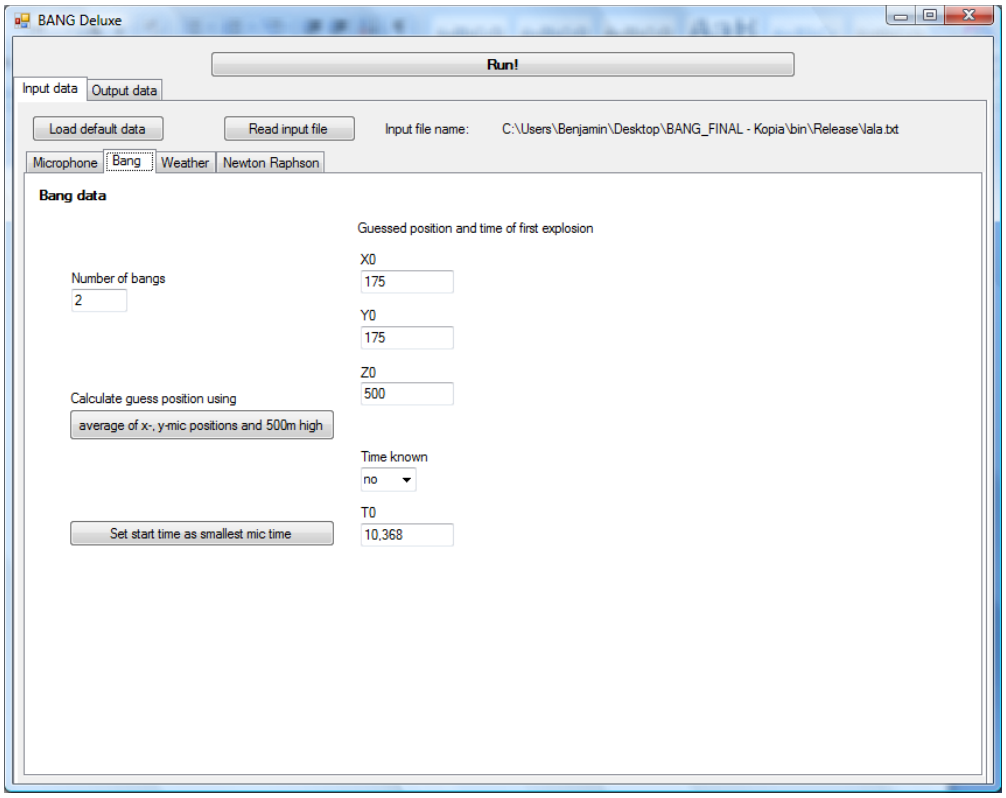
X	Y	Z
175	175	0
50	50	0
50	300	0
300	300	0
300	50	0
0	175	0
175	350	0
350	175	0
175	0	0

Bang time to each microphone

tmic1	tmic2	tmic3	tmic4	tmic5	tmic6	tmic7	tmic8
10,391	10,777	10,397	10,553	10,814	10,629	10,368	10,7
10,513	10,881	10,476	10,595	10,971	10,65	10,435	10,822

(3) *Microphone data tab after having read the input file.*

Bang data



Input file name: C:\Users\Benjamin\Desktop\BANG_FINAL - Kopia\bin\Release\lala.txt

Bang data

Number of bangs: 2

Calculate guess position using: average of x-, y-mic positions and 500m high

Guessed position and time of first explosion

X0: 175

Y0: 175

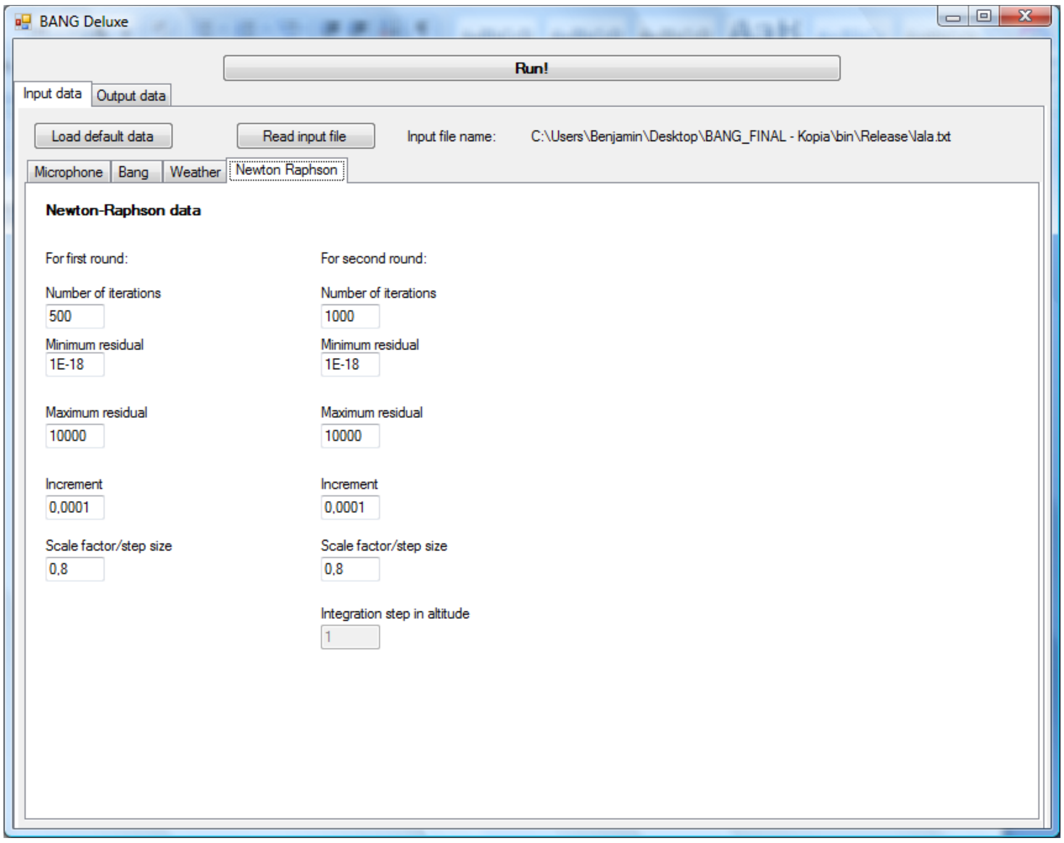
Z0: 500

Time known: no

T0: 10,368

(4) *Bang data tab after having read the input file.*

Newton-Raphson data



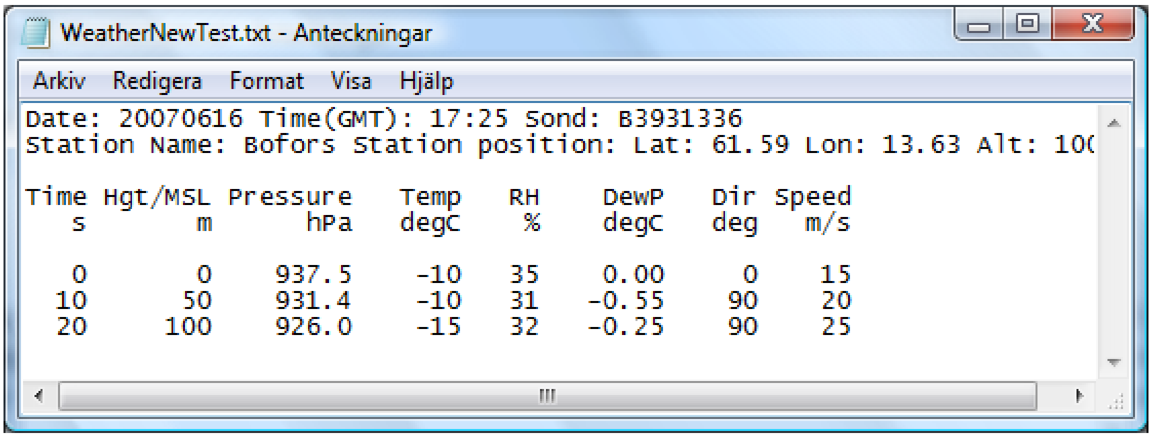
Input file name: C:\Users\Benjamin\Desktop\BANG_FINAL - Kopia\bin\Release\lala.txt

Newton-Raphson data

For first round:		For second round:	
Number of iterations	500	Number of iterations	1000
Minimum residual	1E-18	Minimum residual	1E-18
Maximum residual	10000	Maximum residual	10000
Increment	0,0001	Increment	0,0001
Scale factor/step size	0,8	Scale factor/step size	0,8
		Integration step in altitude	1

(5) *Newton-Raphson data tab after having read the input file.*

Read a weather file: Press ”**Read weather file**” under the weather tab. A weather file is selected in the same way as an input data file. See the opened weather file **(6)**.

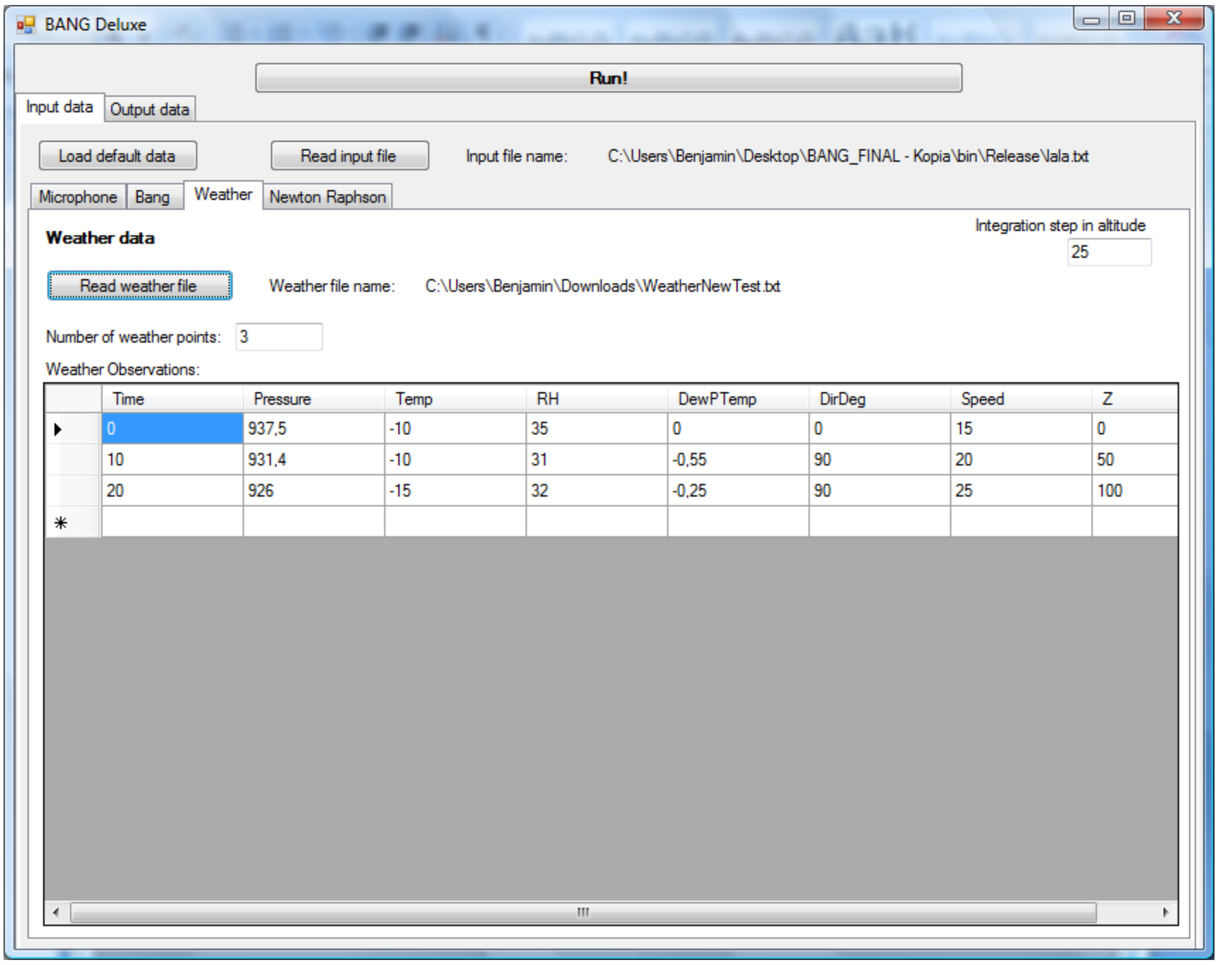


```
Date: 20070616 Time(GMT): 17:25 Sond: B3931336
Station Name: Bofors Station position: Lat: 61.59 Lon: 13.63 Alt: 10(

Time Hgt/MSL Pressure  Temp   RH    DewP   Dir  Speed
   s       m      hPa  degC    %    degC   deg    m/s

   0       0    937.5   -10   35    0.00     0     15
  10      50    931.4   -10   31   -0.55    90     20
  20     100    926.0   -15   32   -0.25    90     25
```

(6) I*nput weather file.*

Because of limited time for this work, only temperature and wind speed and direction are taken into account.

Once the weather file is read, new data will appear in the “Weather data” tab **(7)**.



Input file name: C:\Users\Benjamin\Desktop\BANG_FINAL - Kopia\bin\Release\lala.txt

Weather data

Integration step in altitude: 25

Weather file name: C:\Users\Benjamin\Downloads\WeatherNewTest.txt

Number of weather points: 3

Weather Observations:

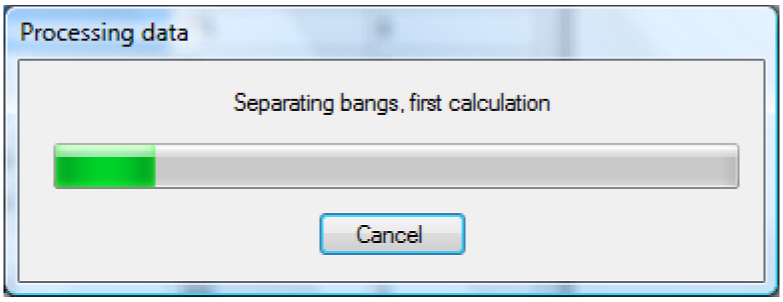
Time	Pressure	Temp	RH	DewPTemp	DirDeg	Speed	Z
0	937,5	-10	35	0	0	15	0
10	931,4	-10	31	-0,55	90	20	50
20	926	-15	32	-0,25	90	25	100

(7) *Weather data tab after having read the weather file.*

The program is now ready to be run once input data file and weather file have been read. Press ”**Run!**” to execute BANG. A ”browse” window will come up asking where you want to save the input data. Select a file or make a new one and click “save”. The calculations will now begin.

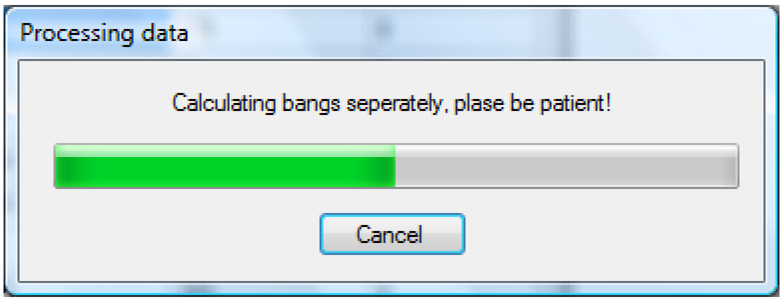
Two output data files are produced, the first part of their name being the name of the saved input data file. The first file contains data from the first calculation regarding discerning the bangs. This file will be named for instance somefilename_output0.txt with _output0.txt on the end. For each new file created, the number at the end of the file name will increment. A second file with the separated bang data will be created named somefilename_output_separated0.txt. This contains the more accurate calculations for each bang.

Once “**Run!**” is pressed and the input data file saved, a progress window with ”Processing data” will pop up **(8)**. The first calculation discerns the bangs.



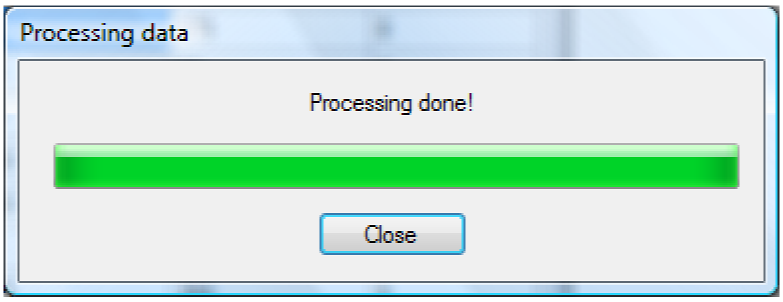
(8). *Progress bar window while separating bangs*

When the first calculation is complete and the second is executed, a new window pops up **(9)**.



(9). *Progress bar window while calculating each bang separately for more accurate results*

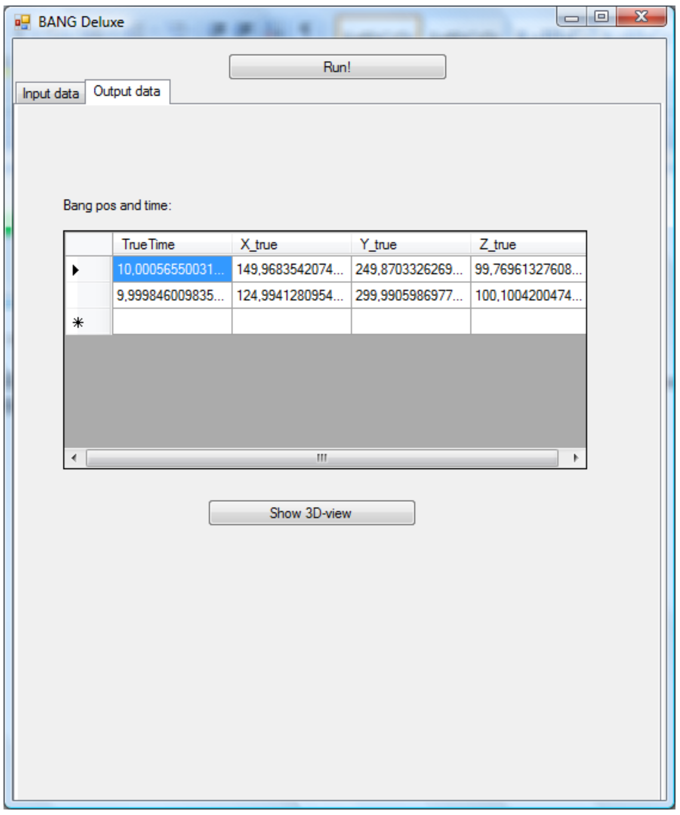
When window **(9)** is up, a little patience is required. It may take a while before the progress bar moves at all. It will progress after each bang has been calculated. When all calculations are complete, a final window emerges **(10)**.



(10). *Final progress window.*

If ”**Cancel**” is pressed during the calculations, the output data files currently being produced will be deleted. A finesse to avoid useless text files taking up disk space and making a mess.

The content of the output data files are not shown here because they are long and take up much space. This is however not necessary since the result can be viewed under the ”**Output data**” tab in the GUI form window **(11)**. The position and time points of the two bangs are shown in the datagridview.

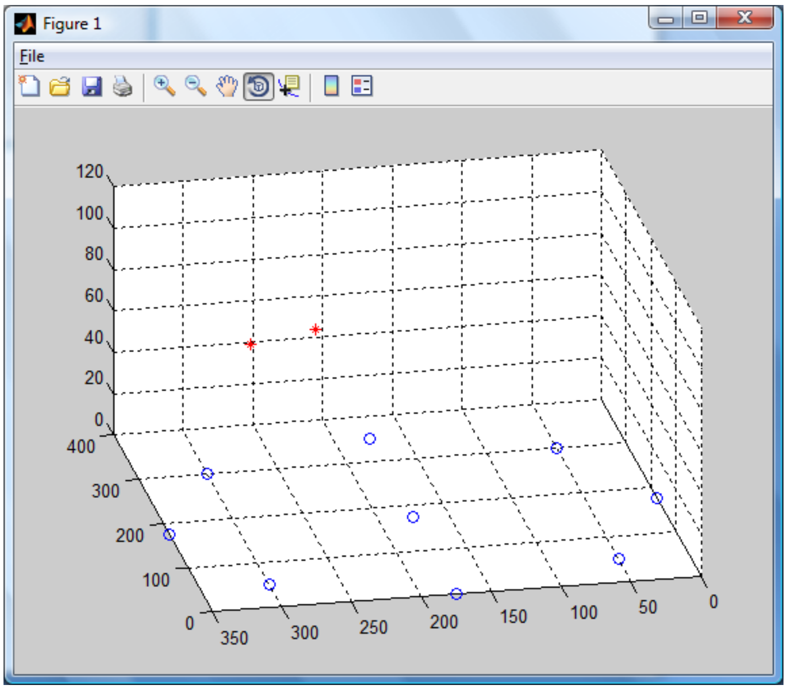


Bang pos and time:

TrueTime	X_true	Y_true	Z_true
10,00056550031...	149,9683542074...	249,8703326269...	99,76961327608...
9,999846009835...	124,9941280954...	299,9905986977...	100,1004200474...

(11) *Result window showing position and time point of bang(s).*

For a 3D-view of the result, press ”**Show input data**”. A Matlab graphics window will appear **(12)**.



(12) *Graphical view of the result, 2 bangs (red), 9 microphones (blue).*