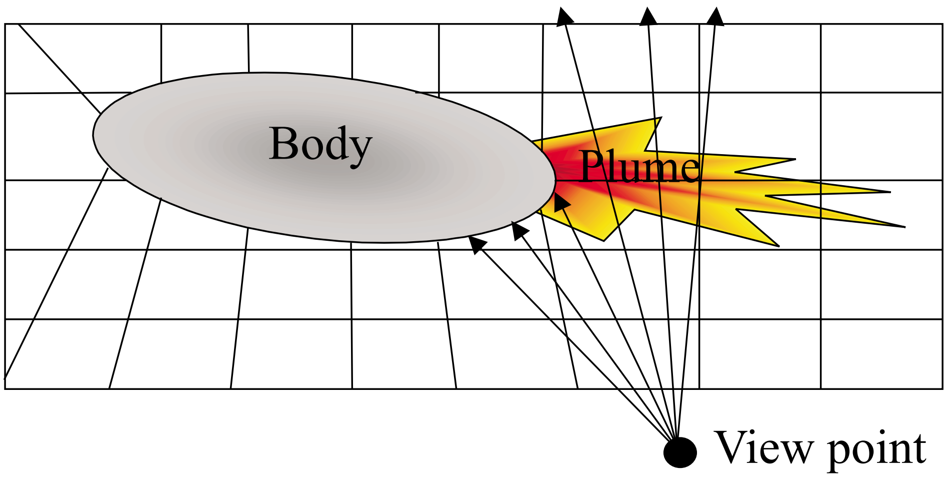
SWEDISH DEFENCE
RESEARCH AGENCY

FOI-R-0554-SE
August 2002
1650-1942

Methodology report

Marlene Andersson and Ingmar Karlsson

SIGGE V1.0β
Users Guide



Division of Aeronautics, FFA
SE-172 90 STOCKHOLM

FOI – Swedish Defence Research Agency
Division of Aeronautics, FFA
SE-172 90 STOCKHOLM

FOI-R-0554-SE
August 2002
1650-1942

Methodology report

Marlene Andersson and Ingmar Karlsson

SIGGE V1.0β
Users Guide

Division of Aeronautics, FFA
SE-172 90 STOCKHOLM

Abstract

A program package, SIGGE, for calculation of IR-signatures, is under development. A detailed description of the β-version is presented in this report together with a short manual, which describes how to run the program.

Contents

1 Introduction

The IR-signature of vehicles has become an important design parameter. The character of the IR-signature will depend on the structure and temperature of the surface of the vehicle. For jet-driven airborne vehicles, the hot plume will also contribute largely to the IR-signature and the composition of the gas will give the signature its characteristics.

For flying objects, the aerodynamic heating and characteristics of the plume can be calculated with Computational Fluid Dynamics, CFD. The CFD-solution can be used as input for calculating the IR-signature with a line-of-sight method [1]. The module based code SIGGE, Figure 1, uses this method to calculate the IR-signature from CFD-solutions. This code, which is written in FORTRAN 90, is under development and this report will describe, in detail, the current structure of SIGGE.

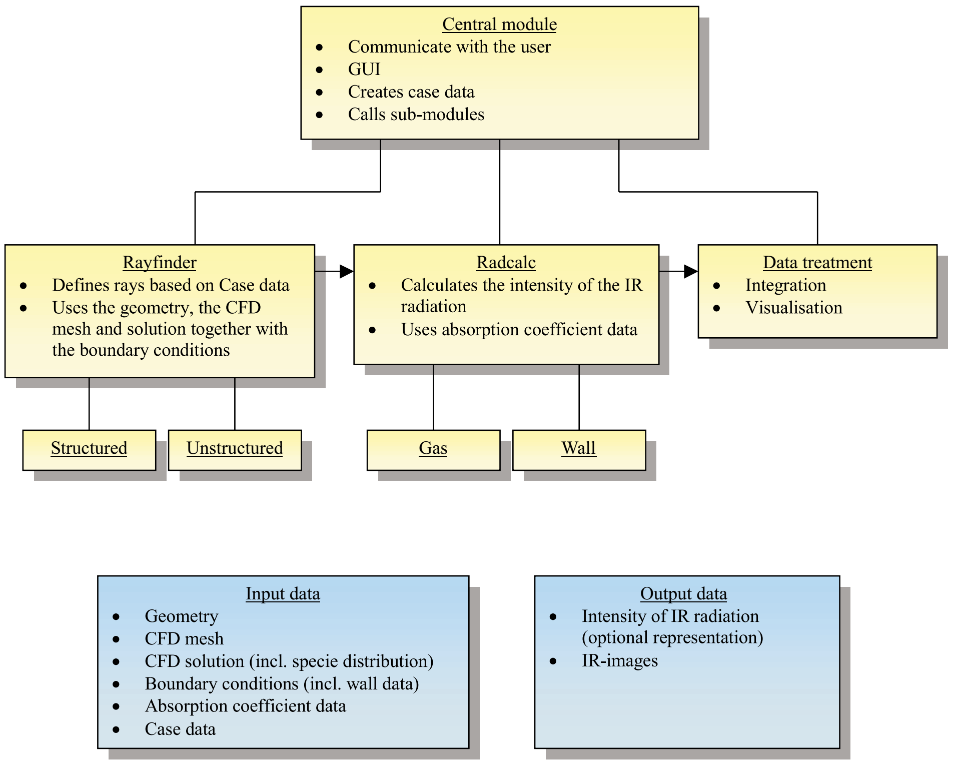

Central module
Communicate with the user
GUI
Creates case data
Calls sub-modules
Rayfinder
Defines rays based on Case data
Uses the geometry, the CFD mesh and solution together with the boundary conditions
Radcalc
Calculates the intensity of the IR radiation
Uses absorption coefficient data
Data treatment
Integration
Visualisation
Structured
Unstructured
Gas
Wall
Input data
Geometry
CFD mesh
CFD solution (incl. specie distribution)
Boundary conditions (incl. wall data)
Absorption coefficient data
Case data
Output data
Intensity of IR radiation (optional representation)
IR-images

Figure 1. The proposed structure of the module based code SIGGE.

1.1 General structure of SIGGE

As seen in Figure 1, the proposed structure of SIGGE contains a central module and three sub-modules. The main purpose of the central module will be to communicate with the user via a graphic user interface (GUI) and to call desired sub-modules. With the GUI, the user will be able to define the scenario wanted.

In the current β-version of SIGGE, a simplified central module exists, where the user can define the scenario by defining parameters in an *input file*, Appendix A, and in a *ray definition file*, Appendix B.

Today, only two of the three sub-modules exist, RAYFINDER and RADCALC. RAYFINDER, presented in section 3, will read the CFD-solution and the mesh, and define rays which are drawn through the mesh and associated with parameters of the CFD-solution. RADCALC, discussed in section 4, will calculate the spectral IR-intensity along these rays using the parameters given by the CFD-solution and a data base with absorption coefficients [1].

Instruction how to run the program, what files that are needed and the file format are presented in section 2. In section 5, the conclusions and an outlook are found.

2 Running SIGGE

2.1 The FFA-format

All files of SIGGE use a common data format, the FFA (Flexible Format Architecture) format [2]. The FFA-data format is self explained, meaning that the data has a name, type and size specification and is identified by its name.

The basic component in the FFA-format is the *data set*. A data set is divided into an identifier and the data. The data set can have one or several sub-data sets, where a sub-data set is a valid data set. In this way, a tree structure is built, see Figure 2.

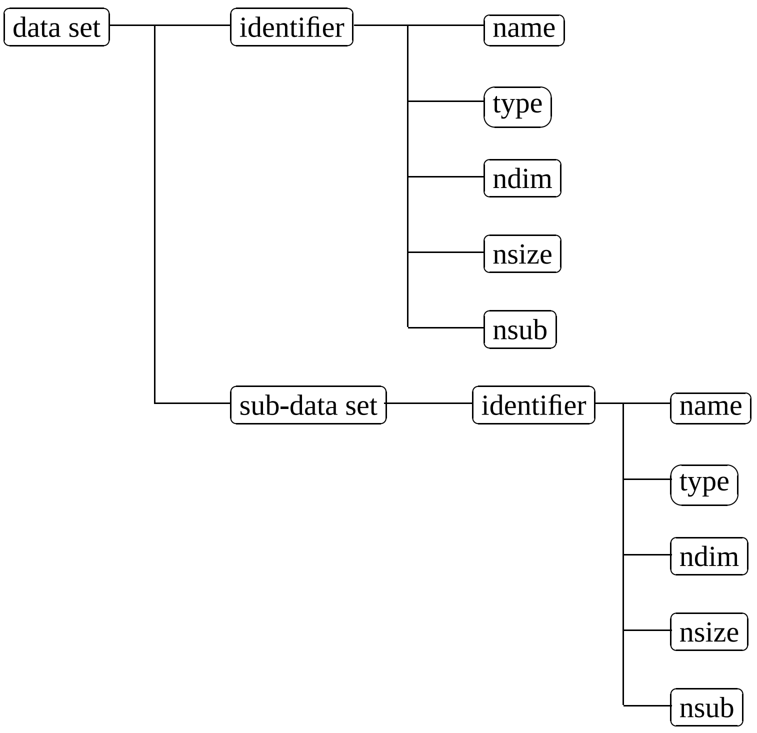


Figure 2. Example of the tree structure of the FFA-data format.

The 5 descriptors are:

name *(character*16)* name of the data set

type *(character*4)* type of the data set given by four characters. The first character defines type of data like integer or real, the second character defines the the type of data set and the third character for field variables will specify where the data is given. See Table 2.1.

ndim *(integer*4)* dimension of data

nsize *(integer*4)* number of data

nsub *(integer*4)* number of sub-data sets

The FFA-format is developed to be used in CFD-programs, thus, some of the types, described in Table 2.1, are specialised for that purpose.

There exists help programs to read the the FFA-format. Such programs are *ffalist* and *ffaJlist*, where *ffaJlist* has a graphic interface requiring JAVA.

Table 1. The different types specified in the FFA-format.

type(1)		type(2)		type(3)	
B	binary*1	I	descriptor	N	node values
I	integer*4	F	field	F	face values
R	real*4	B	boundary information	C	cell centre values
D	real*8	C	constants		
C	complex*(4+4)	T	data tables		
Z	double complex(8+8)	L	data list		
A	character*1				
S	character*16				
L	character*72				
N	no data				

2.2 Input files needed

General input file In this file, variables needed to run the program are defined together with the names of the other input files and the output files. This file is referred to as the *input file* in this report and is described in detail in Appendix A. The file normally has the extension *.ainp*.

Boundary condition file Contains the boundary conditions, used both in the CFD-solution and for generating rays in SIGGE. The extension is *.aboc*.

Mesh file Contains the mesh and has the extension *.bmsh*.

CFD-solution file Contains the CFD-solution with the extension *.bout*.

The *boundary condition file*, *the mesh file* and the *CFD-solution file* have a format which is required by the CFD-solver EDGE [3, 4, 5]. EDGE is an edge-based Euler solver for unstructured grids developed at the Department of Computational Aerodynamics, Aeronautics Division (FFA), FOI.

Ray definition file In this file, the user defines how the rays through the mesh should be drawn. In this report, this file is referred to as the *ray definition file*. The file is described in Appendix B and has the extension *.aray*.

Absorption data base file Contains the absorption coefficients needed to calculate the IR-intensity. The extension is *.btab* and the file is described in Appendix C.

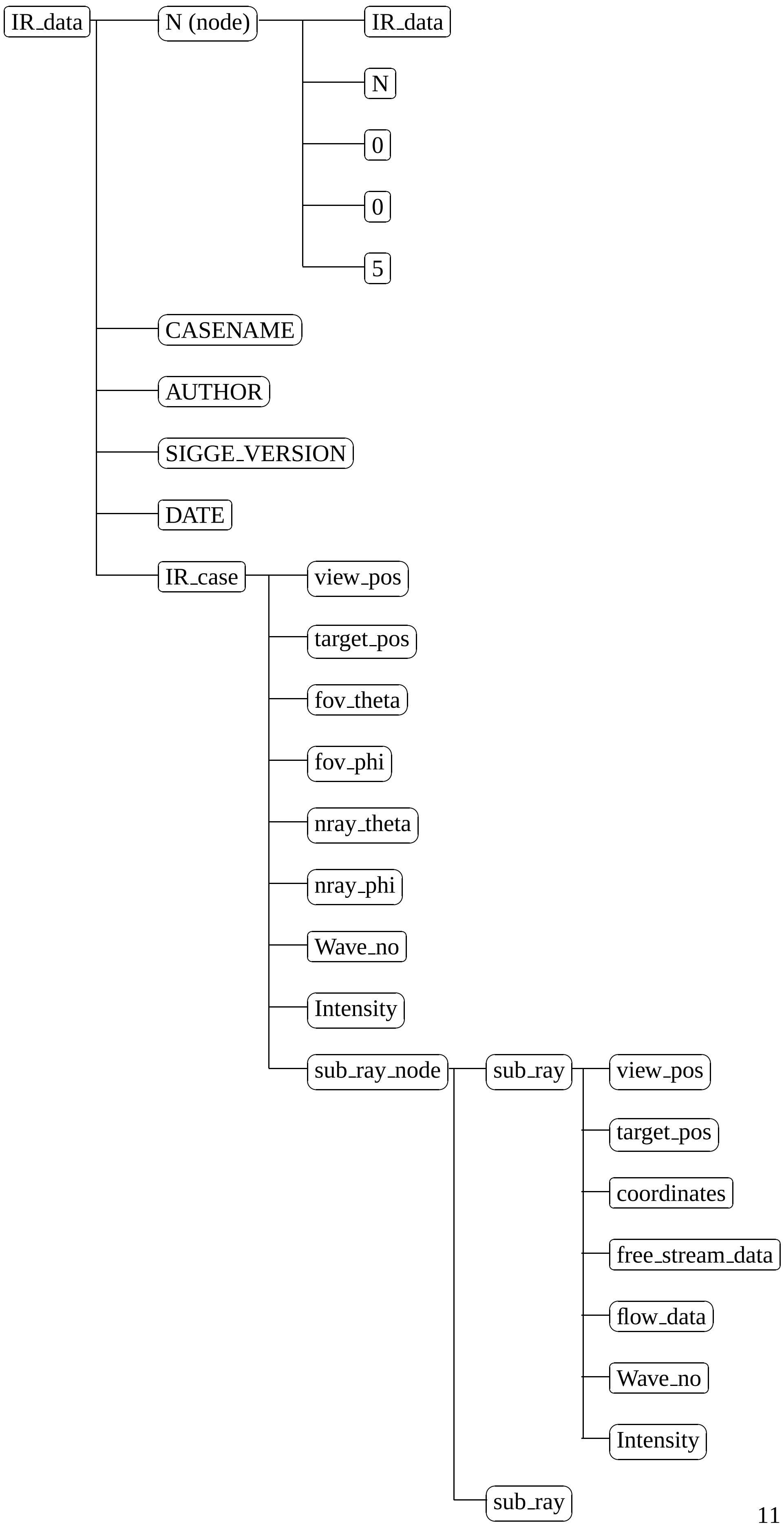
2.3 Output files extracted

Ray data file File generated by the subroutine RAYFINDER, which contains all information needed to calculate the IR-intensity. This is also an input file for the module RADCALC, when discussing this module, this file is referred to as the *indata file*. The extension is *.bdat*

Output file If only RAYFINDER has been run, this file is the same as the *ray data file* described above. If RADCALC has been run, the file has been complemented with the IR-intensities.

The information and structure of the output files are found in Figure 3.

Figure 3. The basic structure of the output files. Except for the data set IR_data, only the names of the different sub-data sets are displayed. The number of sub-data sets, sub_ray, are given by the number of rays defined by the user. The sub-data sets free_stream_data and flow_data contain more sub-data sets.



2.4 To run SIGGE

If SIGGE has been successfully installed, the program will be run with the command:

```
> sigge_1.0 inputfile.ainp
```

for single precision. To run with double precision the following command should be used:

```
> sigge_dble_1.0 inputfile.ainp
```

To reduce calculation errors, double precision is recommended, since a lot of small floating-point numbers are multiplied in the program.

3 RAYFINDER

The program RAYFINDER constructs rays according to the specifications in the *input file*, Appendix A, and in the *ray definition file*, Appendix B. Each ray is discretized by intersections with cells in the CFD-mesh, Figure 4. The intersection

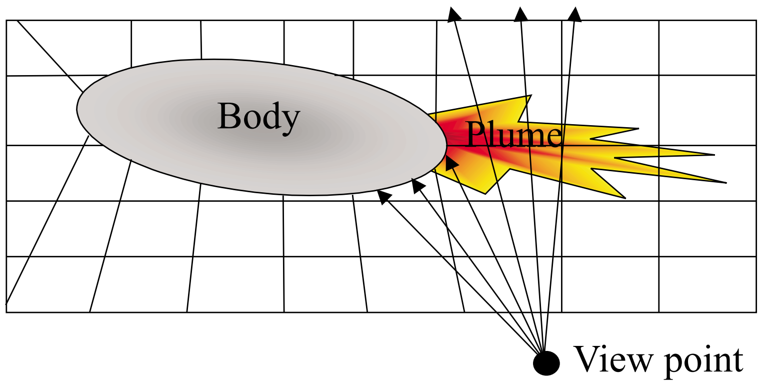


Figure 4. Rays are drawn from a view position toward an object in space through a mesh.

points are then assigned values by interpolating the CFD-solution. The rays are traced through the mesh by an efficient search algorithm.

3.1 Definition of rays

The user will define an imaginary sensor, which consists of a flat plane with a number of pixels, giving an IR-image for each specified wave number, see Figure 5. The orientation of the sensor in space is defined by two orthogonal vectors, $\hat{\theta}$

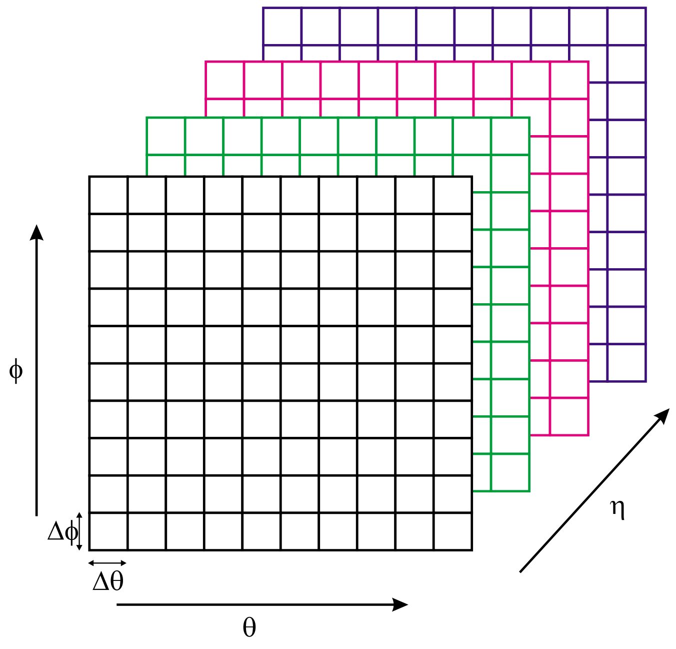


Figure 5. Imaginary sensor defined by the user. The sensor will collect one image for each wave number, η, specified. Each pixel has the field-of-view: $\Delta\theta \times \Delta\phi$.

and $\hat{\phi}$. Each pixel has a field-of-view of $\Delta\theta \times \Delta\phi$, which gives a total field-of-view of $n_\theta\Delta\theta$ and $n_\phi\Delta\phi$, where n_θ and n_ϕ are the number of pixels in the $\hat{\theta}$- and $\hat{\phi}$-directions, respectively.

The user will define a view position, which is the position where the imaginary sensor is located, and a target position, which is the position in space where the centre of the sensor is aiming at, giving an aiming direction, which should be orthogonal to the sensor plane.

The module RAYFINDER will use this information to draw rays from the view position through the mesh. Each ray will be separated from their neighbours with angles of $\Delta\theta$ and $\Delta\phi$. Thus, each ray will correspond to a pixel in the imaginary sensor. In Figure 6, a view and a target position is defined in space, making, in this case, the aiming direction of the sensor run along the z-axis. The imaginary sensor will, therefore, coincide with the x-y plane and the $\hat{\theta}$-direction can be chosen to run along the x-axis and, consistently, the $\hat{\phi}$-direction will run along the y-axis.

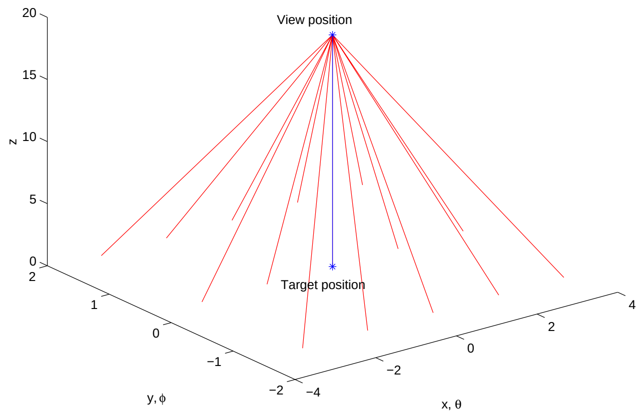


Figure 6. Rays are drawn from a view position where the centre-ray is drawn toward the target position. Each ray is separated from the neighbouring rays with angles of $\Delta\theta$ and $\Delta\phi$.

It is of great importance to point out that the normal spherical coordinates are not used here.

In the *ray definition file*, the user will define the view position, the target position and the orthogonal vectors $\hat{\theta}$ and $\hat{\phi}$ defining the sensor position in space. In the *input file*, the user will define the number of rays in the θ- and ϕ-directions, n_θ and n_ϕ, and the angular resolution, $\Delta\theta$ and $\Delta\phi$.

3.2 Main program of RAYFINDER

The main structure of RAYFINDER is presented in Figure 7. The different subroutines are described in section 3.3. The rays are tracked through the mesh by stepping from cell to cell along the ray. This implies that the implementation is efficient even for grids with a large number of cells. When testing for ray intersections with the cell faces, a line-triangle intersection algorithm, presented in Ref. [6], is used.

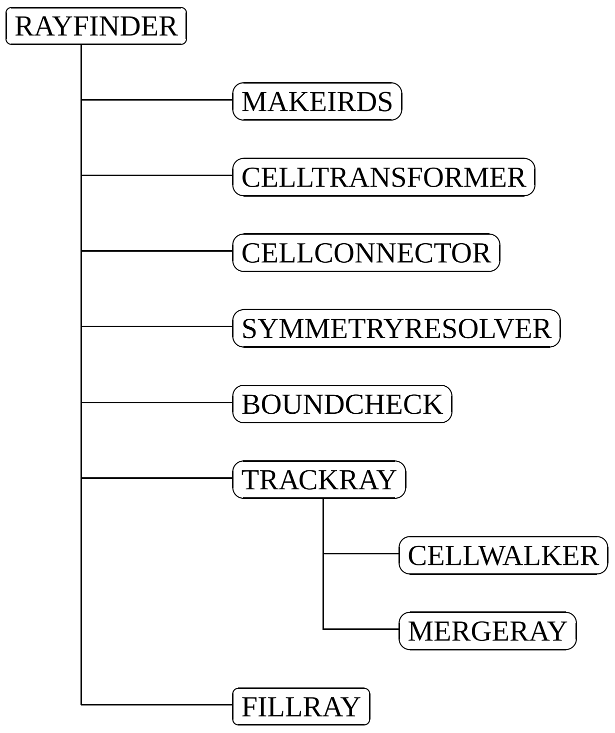


Figure 7. Program structure of RAYFINDER.

3.3 Subroutines

3.3.1 makeirds

The subroutine *makeirds* creates sub rays according to the number of rays requested in the field of view. Each sub ray is assigned an individual target position, giving individual aiming directions, see Figure 6.

3.3.2 celltransformer

The subroutine *celltransformer* transforms all cells in the CFD-mesh to tetrahedra. The following element types are supported: Tetrahedron, hexahedron, prismatic element and pyramid.

3.3.3 cellconnector

The subroutine *cellconnector* stores information for each node, containing element numbers of the elements that the node is a member of. This information is later used in the subroutine *cellwalker*.

3.3.4 symmetryresolver

The subroutine *symmetryresolver* searches all CFD-boundary conditions for symmetry planes. If symmetry planes are found new coordinates are created representing the absent regions. One or two symmetry planes are supported, in the case of two symmetry planes perpendicular planes are required.

3.3.5 boundcheck

The subroutine *boundcheck* searches all CFD-boundaries for ray intersections. The intersections found is used in *trackray* as start and end points when the rays are tracked through the mesh. Additional data such as the direction of the surface normal is calculated and stored for each intersection.

3.3.6 trackray

The subroutine *trackray* handles the tracking of the rays through the mesh. The intersections found in *boundcheck* are used and for each ray attempts are made to find a way through the cells in the mesh that connects the intersections.

3.3.7 cellwalker

The subroutine *cellwalker* is a recursive routine that finds the way through the cells in the mesh given a start point and an end point. Starting in the cell at the start point the end point is searched by step wise walking to one of the neighbouring cells and checking for ray intersections with the new cell.

3.3.8 mergeray

The subroutine *mergeray* cleans up after *cellwalker* and creates the final representation of the rays.

3.3.9 fillray

The subroutine *fillray* interpolates the CFD-solution to the points on the ray, linear interpolation is used. A node based representation of the CFD-solution is required.

4 RADCALC

4.1 Equations used to calculate the IR-intensity

In Ref. [1], the equations needed for calculating the spectral IR-intensity are derived. From that report we get an expression for calculating the spectral transmissivity, τ_η, in the i^{th} cell:

$$\tau_\eta^i = e^{-\kappa_\eta^i s_i} \tag{1}$$

where η is the wave number, κ_η^i is the absorption coefficient and s_i is the length of the i^{th} cell. This simple model is the only available in the current version of SIGGE and the calculation is done in subroutine *calc_tau*, section 4.3.5.

Using the expression in Eq. (1), the spectral intensity, I_η, through a gas is calculated by Eq. (2) in the subroutine *gsimple,* section 4.3.7.

$$\begin{aligned}\mathrm{I}_\eta(gas) = & L_{b\eta}^1(1-\tau_\eta^1)\mathrm{A}_1 + \\ & + L_{b\eta}^2\tau_\eta^1(1-\tau_\eta^2)\mathrm{A}_2 + \\ & + L_{b\eta}^3\tau_\eta^1\tau_\eta^2(1-\tau_\eta^3)\mathrm{A}_3 + \dots\end{aligned} \tag{2}$$

A_i is an area segment defined in Ref. [1]. $\mathrm{L}_{b\eta}$ is the spectral black body radiance given by:

$$\mathrm{L}_{b\eta} = \frac{2\pi h c_0^2 \eta^3}{\pi\left(e^{hc_0\eta/kT}-1\right)} = \frac{C_1\eta^3}{e^{C_2\eta/kT}-1} \tag{3}$$

where

$$C_1 = 2hc_0^2 = 1.191\cdot 10^{-16}\mathrm{Wm}^2 \tag{4}$$

$$C_2 = hc_0/k = 1.4388\cdot 10^{-2}\mathrm{Km} \tag{5}$$

The spectral radiance from a diffuse wall is given by Eq. (6), where ε_η is the emissivity [1] and $\hat{s}_0$ is showing a direction dependence.

$$\mathrm{L}_\eta(\hat{s}_0) = \varepsilon_\eta \mathrm{L}_{b\eta}\tau_\eta^1\tau_\eta^2\tau_\eta^3\dots \tag{6}$$

The spectral intensity from the wall is given by:

$$\mathrm{I}_\eta(wall) = \mathrm{L}_\eta(\hat{s}_0)\times \mathrm{A}_{(TP)} \tag{7}$$

where $\mathrm{A}_{(TP)}$ is the area segment at the wall. This equation is evaluated in subroutine *wsimple,* section 4.3.8.

The total spectral intensity is given by adding the contribution from the gas with the contribution from the wall:

$$\mathrm{I}_\eta = \mathrm{I}_\eta(gas) + \mathrm{I}_\eta(wall) \tag{8}$$

4.2 Main program of RADCALC

The program module RADCALC is a subroutine to SIGGE. It is called with 3 parameters:

pinp *pointer to the general input file*

ptab *pointer to data base with absorption coefficients*

pirds *pointer to indata file (the ray data file described in section 2.3)*

The reader is requested to notice the difference between the *input file* and the *indata file*. The *input file* is presented in Appendix A and the user will define the values in this file. The *indata file* is generated by SIGGE itself, see section 2.3.

In the *input file* and the *indata file*, the pressure should be given in Pa (= N/m^2), temperature in K and wave number in cm^{-1}.

The program structure of RADCALC is presented in Figure 8.

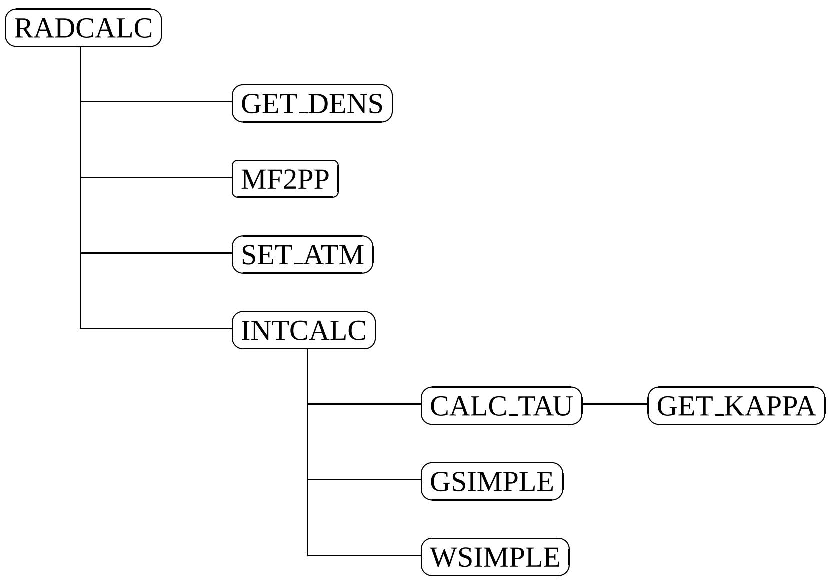


Figure 8. Program structure of RADCALC.

The program module starts with reading the minimum and maximum of wave numbers for a number of intervals together with the resolution from the *input file*. The wave numbers that are going to be used are calculated and added to the pointer *pinp* to make it simple to use in other subroutines. Number of rays as well as the angular resolution are read from the *indata file*.

The concentration of species normally has to be calculated from the mass- and mole fractions, which is done in subroutine *mf2pp*. The mass fraction is defined as the fraction between the ambient air and the combustion gas and the mole fraction as the distribution of species in the combustion gas. It is assumed that the ambient air will not contain species that will contribute to the IR-signature. If no mass fraction is given in the *indata file*, a constant concentration of species can be given in the *input file* and the subroutine *get_dens* will be used instead. The user should define, in the *input file*, whether the mass fraction from the *indata file* or a constant concentration of species should be used.

In the *input file*, the user defines if the free-stream-values or values given in the *input file* should be used for the atmosphere. The subroutine *set_atm* is used to set values for temperature, pressure and density in the atmosphere and to include the distribution of species in the atmosphere.

When all parameters and variables are defined, the program will loop over each ray. The subroutine *intcalc*, which calculates the spectral intensity, is called. The calculated intensity is added to the pointer *pirds* and is written, by SIGGE, to the *output file*, see section 2.3. The resulting spectral intensity has the unit W/sr/cm^{-1}.

4.3 Subroutines

4.3.1 mf2pp

This subroutine calculates the partial density and pressure, for the different species, from the mass fraction given in the *indata file* and mole fraction given in the *input file*.

4.3.2 get_dens

This subroutine defines the partial density and pressure for different species if the mass fraction is not defined in the *indata file*. Constant values given in the *input file* are used.

4.3.3 set_atm

In this routine the distribution of species in the atmosphere is set by using the values defined in the *input file*. For temperature, density and pressure either the free-stream-values or user-defined values are used.

4.3.4 intcalc

Subroutines that could be called from *intcalc* are:

- calc_tau (section 4.3.5)
- gsimple (section 4.3.7)
- wsimple (section 4.3.8)

This subroutine will pick out the coordinates for the current ray and calculate the number of cells along the ray. The coordinates will be transformed from the absolute coordinate system to a one-dimensional system and then the length of each cell is calculated [1].

After this the transmissivity is calculated, section 4.1.

Now one has got everything needed to calculate the spectral intensity. The radiative heat transfer in the gas is calculated as well as the radiation from walls.

4.3.5 calc_tau

For each cell and wave number, the subroutine *get_kappa* is called, which will get absorption coefficients for the different species from the *absorption data base* *file*, section 4.3.6 and Ref. [1]. After this the transmissivity for the i^{th} cell will be calculated according to:

$$\tau^i_\eta = \prod_{sp} e^{\kappa^{sp}_\eta s_i} \tag{9}$$

where sp stands for different species: e.g. *CO_2, H_2O, Particles, HCl* and *CO*.

4.3.6 get_kappa

This subroutine will read the absorption coefficients from the *absorption data base file*, Appendix C. The absorption coefficients are either given as pressure absorption coefficients, $\kappa^{sp}_{p\eta}$, or mass absorption coefficients, $\kappa^{sp}_{\rho\eta}$. These absorptions coefficients have to be multiplied with either the partial pressure or the partial density:

$$\kappa^{sp}_\eta = \kappa^{sp}_{p\eta} \times p_{sp} \tag{10}$$

$$\kappa^{sp}_\eta = \kappa^{sp}_{\rho\eta} \times \rho_{sp} \tag{11}$$

The units for $\kappa^{sp}_{p\eta}$ should be (atm $\times$ cm)$^{-1}$ and for $\kappa^{sp}_{\rho\eta}$ it should be m$^3\times$(kg $\times$ cm)$^{-1}$. It has to be given in the *input file* whether the data base contains pressure absorption coefficients or mass absorption coefficients.

The program will check if the temperature and wave number of interest is in the interval of the data base, if not the closest one will be used.

The program will use bilinear interpolation to find the absorption coefficients in the data base.

The partial pressure of interest or the partial density of interest will be found and multiplied with either the pressure absorption coefficients or the mass absorption coefficients as described above. Absorption coefficients for the different species specified will be returned to the subroutine *calc_tau*.

4.3.7 gsimple

This subroutine is used to calculate the heat transfer in a gas using a simple model. The wave number is transformed from cm^{-1} to m^{-1} and the spectral intensity is calculated with Eq. (2).

4.3.8 wsimple

Here the outgoing intensity from a wall is calculated as described in Eq. (7). The emissivity is defined in the *input file* and is assumed to be constant over the whole wall in this simple model.

5 Conclusions and outlook

The present version of SIGGE is capable of calculating the spectral intensity from a gas and of diffuse walls, using simple equations. In this report the general structure of the module based program has been presented. Files needed and extracted have been discussed. All parts of the current version of SIGGE have been described.

A validation of SIGGE has been performed for the gas part, presented in Ref. [7], with satisfactory result. The development of SIGGE will continue and in future versions of SIGGE the following features will be added:

- Ability to handle reflections in walls (will be implemented shortly)
- Calculation of both spectral intensity and spectral radiance
- Ability to use either mass fraction (already implemented) or mole fraction (will be implemented as soon as possible) for the distribution of species
- More complex equations for specific species
- Ability to handle non-diffuse surfaces
- Ability to handle atmospheric background
- A user friendly interface
- Data treatment

References

[1] M. Andersson, *Derivation of Equations used by the Computer Program SIGGE for Calculating IR-intensity*, FOI-R-0553-SE, (2002).

[2] S. Wallin, *Standardized Data Format, version 1*, FFAP-A-950, (1992).

[3] T. Sjögren and P. Eliasson, *Description and Validation of EDGE*, FFA TN 1998-61, (1998).

[4] M. Sillén and A. Karlsson, *Användarhandledning Spider/Tritet/Edge*, FFO-2001-0062, (2001).

[5] *EDGE V2.3 GUIDE*, internal users guide at the Department of Computational Aerodynamics, Aeronautics Division (FFA), FOI, (2002).

[6] R. Löhner and P. Parikh, *Generation of three-dimensional unstructured grids by the advancing-front method*, International Journal for Numerical Methods in Fluids, vol. 8, pp.1135-1149, (1988).

[7] M. Andersson, *Validation of the Computer Program SIGGE against Spectral IR-measurements on an Engine Test Rig*, FOI-R-0555-SE, (2002).

Appendix A

The input file

The *input file* is common for the modules RAYFINDER and RADCALC. In this file, some parameters are given to define different cases or scenarios. The input and output files are also given in this file.

Example of how a parameter should be defined is described in Figure 9.

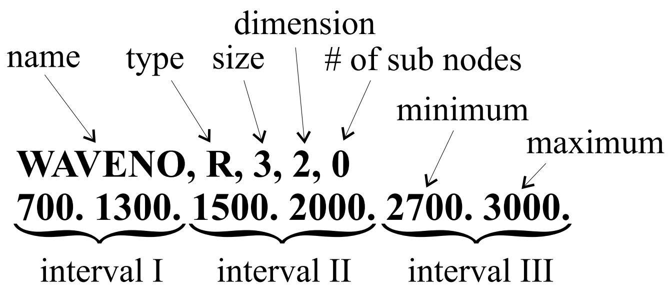


Figure 9. Example of the definition of one parameter in the *input file*. Here is the parameter WAVENO defined, which is of type *REAL*, has size 3 and dimension 2. In this case the number of intervals is defined with the size.

General parameters set in the *input file* are:

SIGGE_VERSION *Program version*

CASENAME *Case description name*

AUTHOR *work done by. . .*

For RAYFINDER, the following parameters are defined:

RUNRAYFINDER *if RAYFINDER should be ran or not*

- 0: No
- 1: Yes

FOV_THETA *field of view in the θ-direction (degrees)*

FOV_PHI *field of view in the ϕ-direction (degrees)*

NRAY_THETA *number of rays per viewpoint in the θ-direction*

NRAY_PHI *number of rays per viewpoint in the ϕ-direction*

For RADCALC, the following parameters are defined:

RUNRADCALC *if RADCALC should be ran or not*

- 0: No
- 1: Yes

WAVENO *min and max of wave number, number of intervals should also be given*

DWAVENO *resolution which will define the number of wave numbers*

DATABASE_TYPE *the type or unit of the absorption coefficients*

- 1: mass absorption coefficients
- 2: pressure absorption coefficients

TRANSMODEL *the transmissivity model*

- 1: Simplest model, same for all gases

GASMODEL *type of gas model*

- 0: No gas model (gives intensity: I$_\eta(gas)$ = 0)
- 1: Basic gas model

SPNAME *list of species, order should be the same as in the data base of absorption coefficients*

SPTYPE *gives on which form the distribution of species is given*

- 0: No distribution of species given in the *indata file*
- 1: Mass fraction given in *indata file*
- 2: Mole fraction of species given in *indata file* (not implemented)

SPDIS *distribution of species in gas if not given in the indata file (only if* SPTYPE *set to 0)*

MOLEFRAC *needed to convert the mass fraction to pressure and density distribution of the species (only if* SPTYPE *is set to 0)*

ATMVAL *which temperature and pressure values should be used in the atmosphere*

- 0: Use free-stream-values
- 1: Use values set by ATM

ATM *temperature [K] and pressure [Pa] in the atmosphere, if free-stream-values not are used (i.e.* ATMVAL *is set to 1)*

SPATM *distribution of species in the atmosphere*

WALLMODEL *type of wall model*

- 0: No wall model (gives intensity: I$_\eta(wall)$ = 0)
- 1: Basic wall model

EMISSIVITY *emissivity for walls*

ATMOMODEL *type of model for atmospheric background*

- 0: No atmosphere model (gives intensity: I$_\eta(atmosphere)$ = 0)
- 1: Basic atmosphere model (not implemented)

Input and output files defined in the *input file*:

BCFILE *Boundary condition file*

MSHFILE *Mesh file*

SOLFILE *CFD-solution file*

RAYFILE *Ray definition file*

ABSFILE *Absorption coefficient data base file*

RAYTEMPFILE *Ray data file. Contains information about the rays, including densities, temperature and such things. This file must exist before RADCALC starts, if it does not exists, it must be extracted by RAYFINDER, i.e run RAYFINDER (set option in inputfile).*

OUTFILE *Output file with wave number, intensity and other things, see section 2.3.*

In the Tables 2, 3, 4 and 5, the descriptors for the parameters defined in the *input file* are described.

Table 2. The descriptors for the general parameters set in the *input file*.

name	type	size	dimension	sub nodes
SIGGE	N	0	0	32
SIGGE_VERSION	SI	1	1	0
CASENAME	LI	1	1	0
AUTHOR	LI	1	1	0

Table 3. The descriptors for RAYFINDER set in the *input file*.

name	type	size	dimension	sub nodes
RUNRAYFINDER	I	1	1	0
NRAY_THETA	I	1	1	0
NRAY_PHI	I	1	1	0
FOV_THETA	I	1	1	0
FOV_PHI	I	1	1	0

Table 4. The descriptors for RADCALC set in the *input file*.

name	type	size	dimension	sub nodes
RUNRADCALC	I	1	1	0
WAVE_NO	R	# of intervals	2	0
DWAVENO	I	# of intervals	1	0
DATABASE_TYPE	I	1	1	0
TRANSMODEL	I	1	1	0
GASMODEL	I	1	1	0
SPNAME	L	1	# of species	0
SPDIS	R	1	# of species	0
FIMAX	I	1	1	0
MOLEFRAC	R	1	# of species	0
ATMVAL	I	1	1	0
ATM	I	1	2	0
SPATM	R	1	# of species	0
WALLMODEL	I	1	1	0
EMISSIVITY	R	1	1	0
ATMOMODEL	I	1	1	0

Table 5. The descriptors for the file names set in the *input file*.

name	type	size	dimension	sub nodes
BCFILE	L	1	1	0
MSHFILE	L	1	1	0
SOLFILE	L	1	1	0
RAYFILE	L	1	1	0
ABSFILE	L	1	1	0
RAYTMPFILE	L	1	1	0
OUTFILE	L	1	1	0

Appendix B

The ray definition file

The sensor position and the spatial orientation of the sensor is specified by the user in the *ray definition file*. An arbitrary number of sensor cases can be used and for each case a sub-data set is defined. For each case the following parameters must be specified:

VIEWPOS *coordinates of the sensor position*

TARGETPOS *coordinates of a point on the sensors line of sight in the centre of the field of view*

THETADIR *vector defining theta direction*

PHIDIR *vector defining phi direction*

In Table 6 is the descriptors described.

Table 6. The descriptors for parameters set in the *ray definition file*.

name	type	size	dimension	sub nodes
RAYDATA	N	0	0	# of ray sets
RAY	N	0	0	4
VIEWPOS	RD	3	1	0
TARGETPOS	RD	3	1	0
THETADIR	RD	3	1	0
PHIDIR	RD	3	1	0

Appendix C

The absorption data base file

To be able to calculate the IR-intensity, the absorption coefficients, κ_η, have to be known, see Eq. (1). The absorption coefficients are different for different species and are dependent on the temperature and wave number. Thus, in the *absorption data base file,* the different species have their own sets of absorption coefficients, which are connected to temperature and wave number. The structure of the file is presented in Figure 10.

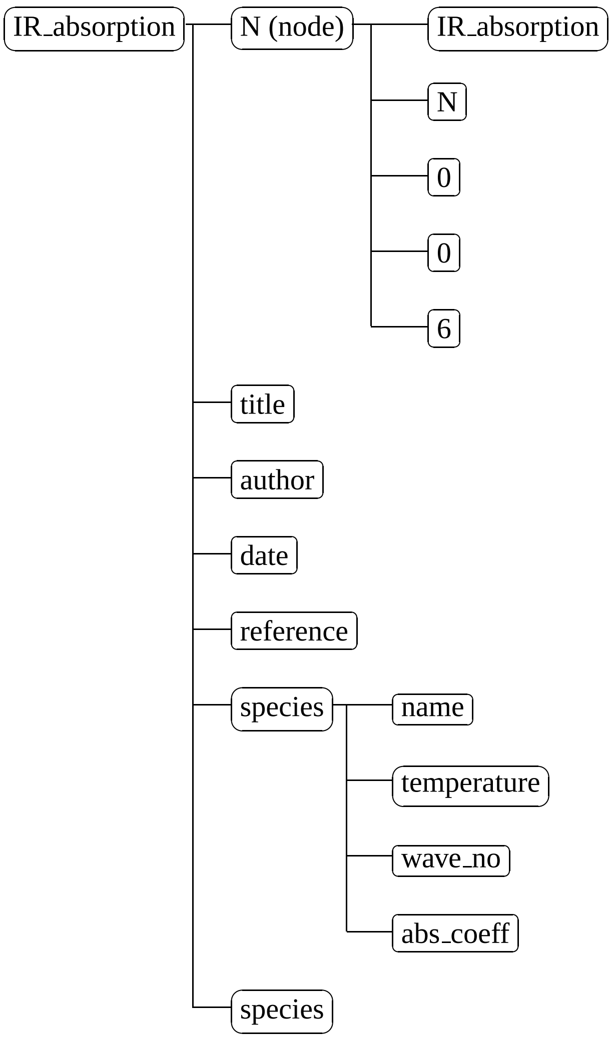


Figure 10. The structure of the absorption data base file.

Issuing organisation FOI – Swedish Defence Research Agency Division of Aeronautics, FFA SE-172 90 STOCKHOLM	Report number, ISRN FOI-R-0554-SE	Report type Methodology report
	Month year August 2002	Project number E840346
	Customers code 3. Aeronautical Research	
	Research area code 7. Vehicles	
	Sub area code 73. Aeronautical Research	
Author(s) Marlene Andersson and Ingmar Karlsson	Project manager Marlene Andersson	
	Approved by Bengt Winzell Head, Computational Aerodynamics Department	
	Scientifically and technically responsible Marlene Andersson	
Report title SIGGE V1.0β *Users Guide*		
Abstract A program package, SIGGE, for calculation of IR-signatures, is under development. A detailed description of the β-version is presented in this report together with a short manual, which describes how to run the program.		
Keywords IR-signature, SIGGE, documentation		
Further bibliographic information		
ISSN 1650-1942	Pages 35	Language English
	Price Acc. to price list	
	Security classification Unclassified	

Utgivare Totalförsvarets Forskningsinstitut – FOI Avdelningen för Flygteknik, FFA SE-172 90 STOCKHOLM	Rapportnummer, ISRN FOI-R-0554-SE	Klassificering Metodrapport
	Månad år Augusti 2002	Projektnummer E840346
	Verksamhetsgren 3. Flygteknisk forskning	
	Forskningsområde 7. Bemannade och obemannade farkoster	
	Delområde 73. Flygteknisk forskning	
Författare Marlene Andersson och Ingmar Karlsson	Projektledare Marlene Andersson	
	Godkänd av Bengt Winzell Chef, Institutionen för beräkningsaerodynamik	
	Tekniskt och/eller vetenskapligt ansvarig Marlene Andersson	
Rapporttitel SIGGE V1.0β *Användarhandledning*		
Sammanfattning Ett programpaket, SIGGE, för beräkning av IR-signaturer är under utveckling. En detaljerad beskriving av en β-version presenteras i den här rapporten tillsammans med en kort manual som beskriver hur programmet körs.		
Nyckelord IR-signatur, SIGGE, dokumentation		
Övriga bibliografiska uppgifter		
ISSN 1650-1942	Antal sidor 35	Språk Engelska
Distribution enligt missiv	Pris Enligt prislista	
	Sekretess Ej hemlig	