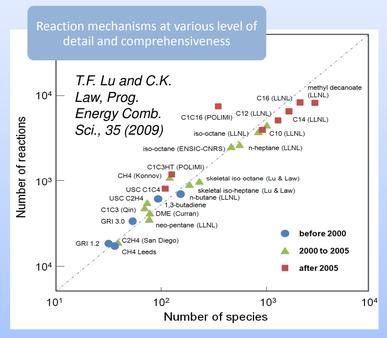
XXXIV Evento della Sezione Italiana del Combustion Institute, 24th - 26th October 2011, Rome

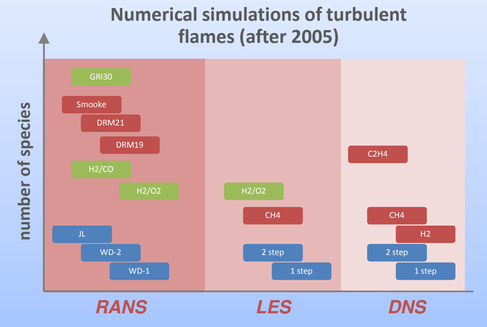
OpenSMOKE: Numerical modeling of reacting systems with detailed kinetics

A. Cuoci, A. Frassoldati, T. Faravelli, E. Ranzi
alberto.cuoci@polimi.it
Department of Chemistry, Materials and Chemical Engineering, Politecnico di Milano
Piazza Leonardo da Vinci 32, 20133, Milano, Italy

Motivation



Today it is well recognized that realistic numerical simulations of combustion phenomena must necessarily require not only an accurate, detailed description of fluid dynamic aspects, but also a realistic characterization of the chemistry and the physical and chemical properties of the gas mixtures involved. In the last years the inaccuracy of simplistic descriptions assuming either equilibrium chemistry or global mechanisms in reacting flows has been clearly demonstrated. At the same time there has been an increasing effort to incorporate more complex reaction mechanisms in simulation of combustion processes and this has led to the development of reaction mechanisms with different levels of detail and comprehensiveness. Kinetic mechanisms sufficiently realistic and comprehensive usually consist of a large number of species and reactions. As a consequence the computational cost associated with such mechanisms is usually very high. Moreover, chemical species are not only are non-linearly coupled, but frequently they have different time scales, which make the simulations computationally stiff. Hence there is the need of computational tools to make computationally efficient the management of large kinetic schemes and easy their integration in new and/or existing numerical codes.
In this work the **OpenSMOKE** framework is presented, a collection of C++ libraries specifically conceived to manage large, detailed kinetic schemes (with hundreds of species and thousands of reactions) in numerical simulations of reacting flows.



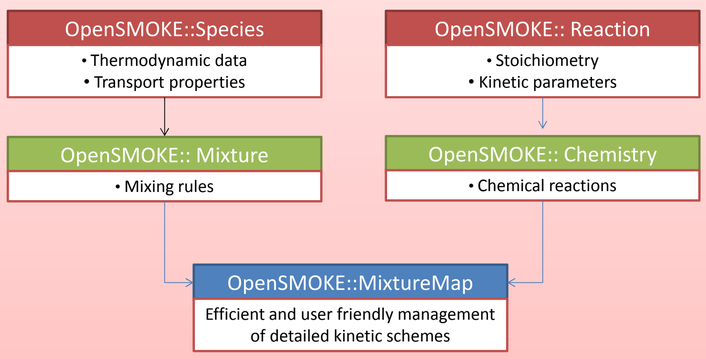
Methodology

Object oriented Programming (OOP)

The **OpenSMOKE** framework is based on Object-Oriented Programming (OOP), since it is generally recognized that OOP produces code that is easier to write, validate, and maintain than procedural techniques. The OOP approach involves three main features: abstraction, inheritance, and polymorphism. ***Abstraction*** is the ability to represent conceptual constructs in the program and to hide details behind an interface. This is achieved by the concept of *classes* to represent conceptual objects in the code, that encapsulate (i.e. contain and protect) the data that make up the object. Member functions are provided that permit limited, well-defined access to the encapsulated data. ***Inheritance*** enables relationships between the various classes to be expressed, representing commonality between different classes of objects by class extension. ***Polymorphism*** is the ability to provide the same interface to objects with different implementations, thus representing a conceptual equivalence between classes that in practical terms have to be coded differently. The **OpenSMOKE** library is written in **C++**, since this is a good programming language for scientific work. It is widely available on all platforms and, being based on C, is fast. Several studies indicate no significant difference in performance between Fortran and the C group of languages

Policies

The **OpenSMOKE** library is based on ***template*** programming and strongly relies on the concept of ***policies*** and **policy classes**, an important class design technique that enable the creation of flexible, highly reusable libraries. In brief, policy-based class design fosters assembling a class with complex behavior out of many little classes (called policies), each of which takes care of only one behavioral or structural aspect. A policy establishes an interface pertaining to a specific issue. The user can implement policies in various ways as long as the policy interface is respected. Since policies can be mixed and matched, a combinatorial set of behaviors can be achieved by using a small core of elementary components.



User-friendly interface

```
1 OpenSMOKE::Mixture *mix = new OpenSMOKE::Mixture(fileName);
2 OpenSMOKE::MixtureMap *map = mix->CreateMap<0>();

3 map.updateStatus(T,P,y);              // update status
4 cp = map->cp();                       // specific heats
5 viscosityMix = map->viscosityMix();   // mix viscosity
6 R = map->formationRates();            // formation rates
7 r = map->reactionRates();             // reaction rates
```

Efficiency

- ***MixtureMap:*** a specifically conceived class is used for calculating thermodynamic, transport properties and kinetics data in a fast way, without using complex interfaces. From every Mixture object (working as a sort of C++ Factory) a number of independent MixtureMap objects can be created. Then, when some property is needed, the user has only to update the status of the MixtureMap object and ask for such specific property
- ***code reformulation:*** many parts of the numerical algorithms are reformulated in a less intuitive way in order to minimize the number of flops needed to perform some calculations or to avoid the usage of CPU-expensive functions. Moreover, the OpenSMOKE code tries always to handle objects efficiently, using only pass-by-reference and return-by-reference techniques, without over-using costly language features (such as exceptions, virtual methods and RTTI) and strongly applying inline methods;
- ***caching:*** the code is written in order to cache as much as possible, which means storing items for future use in order to avoid retrieving or recalculating them. Only calculations which are strictly necessary are performed on the fly. If some variables can be calculated only once, they are stored so that they are available for future needs;
- ***object pools:*** they are a technique for avoiding the creation and deletion of a large number of objects during the code execution. If the user knows that his code needs a large number of short-lived objects of the same type, he creates a pool of those objects. Whenever he needs an object in his code, he asks the pool for one. When the user is done with the object, he returns it to the pool. The object pool creates the objects only once, so their constructor is called only once, not each time they are used;
- ***optimized functions:*** the numerical algorithms are often reformulated in order to exploit the Intel® MKL Vector Mathematical Functions Library (VML). VML includes a set of highly optimized functions (arithmetic, power, trigonometric, exponential, hyperbolic, special, and rounding) that operate on vectors of real and complex numbers;

Coupling with external libraries

OpenSMOKE Suite

The **OpenSMOKE** library was coupled to the **BzzMath** libraries [11] (freely available at http://homes.chem.polimi.it/gbuzzi/) to solve reacting systems which are typically studied for kinetic purposes (ideal reactors, laminar opposed flames, laminar premixed flat flames, flamelets, etc.)

OpenSMOKE Suite
Collection of C++ codes for the numerical simulation of reacting systems
• ideal reactors (PSR, plug-flow, batch, shock-tube)
• laminar counter flow diffusion flames
• premixed flat flames
• laminar flamelets

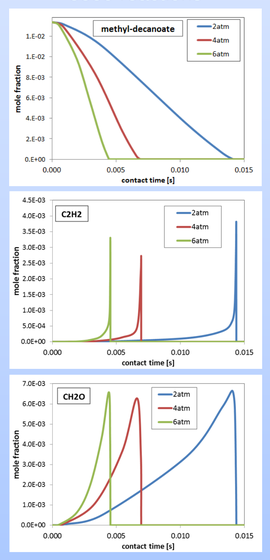
Numerical Libraries
BzzMath
• linear algebra
•non linear systems, ODE, DAE
Additional libraries
• sparse linear solvers
• eigenvalues

OpenSMOKE Framework
Complex Gas-Phase Chemistry
• homogeneous reactions
• detailed transport properties
• CHEMKIN compatible
• ~100 species, ~1000 reactions
• Sensitivity Analysis, ROPA, etc.

The **OpenSMOKE** Suite is released as a collection of several solvers, devoted to specific reacting systems. Detailed kinetic schemes can be written in CHEMKIN format. The **OpenSMOKE** Suite was successfully used for the numerical modeling of premixed flat laminar flames, laminar counter flow diffusion flames, diffusion flames from liquid pools, unsteady and oscillating laminar flames, etc.

Adiabatic PFR

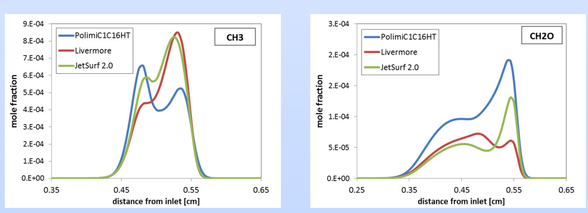
Methyl-Butanoate Mechanism
Herbinet et al. (2009)
2878 species
8555 reactions



T = 1000K
P = 2, 4, 6 atm
Feed: MB/Air (ϕ=1)

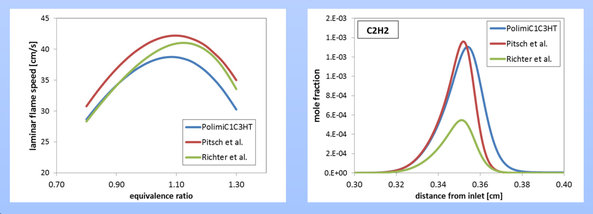
Laminar counter flow flames

n-C_7H_{16}/Air @ T=300K, P=1atm
PolimiC1C6HT: 167 species, 4744 reactions
Livermore.: 654 species, 5264 reactions
JetSurf 2.0.: 344 species, 2163 reactions



Premixed flat laminar flames

C_3H_8/Air @ T=300K, P=1atm
PolimiC1C6HT: 178 species, 4744 reactions
Pitsch et al.: 158 species, 1049 reactions
Richter et al.: 158 species, 872 reactions



laminarSMOKE

The OpenSMOKE library was coupled to the **OpenFOAM®** framework to create a new solver, called **laminarSMOKE**, for the numerical simulation of multidimensional laminar flames

OpenFOAM® Framework
Complex 2D/3D geometries
• unstructured meshes
• complex boundary conditions
• state of the art schemes for discretisation
art schemes for discretisation

laminarSMOKE
CFD code for laminar reacting flows with detailed kinetics
• completely open source
• easily extensible (object oriented design)
• efficient

OpenSMOKE Framework
Complex Gas-Phase Chemistry
• homogeneous reactions
• detailed transport properties
• CHEMKIN compatible
• ~100 species, ~1000 reactions
• Sensitivity Analysis, ROPA, etc.

References

[1] **Weller H. G., Tabor G., Jasak H., Fureby C.,** *Computers in Physics* 12: 620-631 (1998)
[2] **Stroustrup B.,** *The C++ Programming Language, 3rd Edition*, Addison–Wesley, Reading (MA), 1997
[3] **Cary J. R., et al.,** *Computational Physics Communications* 105: 20-36 (1997)
[4] **Alexandrescu A.**, *Modern C++ Design: Generic Programming and Design Patterns Applied*, Addison-Wesley, (2001)
[5] **Smooke M. D., Rabitz H., Reuven Y., Dryer F. L.,** *Combustion Science and Technology* 59: 295-319 (1983)
[6] **Herbinet O., Pitz W. J., Westbrook C. K.,** *Combustion and Flame* 154: 507-528 (2008)
[7] **Ranzi E., Frassoldati A., Granata S., Faravelli T.,** *Industrial and Engineering Chemistry Research* 44: 5170-5183 (2005)
[8] **Narayanaswamy K., Blanquart G., Pitsch H.,** *Combustion and Flame* 157: 1879-1898 (2010)
[9] **Richter H., Howard J. B.,** *Physical Chemistry Chemical Physics* 4: 2038-2055 (2002)
[10] **Marinov N. M., et al.,** *Combustion and Flame* 114: 192-213 (1998)
[11] **Buzzi-Ferraris G.,** *BzzMath 6.0 Numerical Libraries*, 2011
[12] **Cuoci A., Frassoldati A., Faravelli T., Ranzi E.,** *Combustion and Flame* 156: 2010-2022 (2009)
[13] **Grana R., et al.,** *Combustion Theory and Modelling* In Press (2011)
[14] **OpenCFD Ltd.,** *OpenFOAM®* http://wwwopenfoamorg/

CRECK modeling research group - Chemical Reaction Engineering and Chemical Kinetics
Politecnico di Milano
http://creckmodeling.chem.polimi.it