Welcome to the First Workshop on Cantera!
Timeline

1997: DARPA/NSF Virtual Integrated Prototyping Program
1998: Earliest private versions of Cantera
1999: 1st Cantera Workshop
2000: User's group established
2001: SourceForge
2002: MixMaster released
2003: 1D Flames
2004: MATLAB Toolbox

Transport Properties
Preregistered Workshop Attendees

- UC Berkeley: 19%
- UIUC: 13%
- Caltech: 9%
- DOE/NETL: 6%
- Comb. Sci. Eng.: 3%
- Sandia: 3%
- Berlin U. Tech. (Germany): 3%
- U. Heidelberg (Germany): 3%
- U. Windsor (Canada): 3%
- Swiss Fed. Inst. Tech. (Switzerland): 3%
- Telemark U. (Norway): 3%
- U. Pittsburgh: 3%
- U. Utah: 3%
- ANU (S. Korea): 3%
- Colo. Sch. Mines: 3%
- Aristotle U. (Greece): 3%
- Yale: 3%
- TU Denmark: 6%
<table>
<thead>
<tr>
<th>Time</th>
<th>Speaker(s)</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>9:00</td>
<td>D. Goodwin</td>
<td>Introduction to Cantera</td>
</tr>
<tr>
<td>9:20</td>
<td>D. Goodwin</td>
<td>Defining and Using Phases and Interfaces</td>
</tr>
<tr>
<td>9:50</td>
<td>D. Goodwin</td>
<td>Reactor Networks</td>
</tr>
<tr>
<td>10:15</td>
<td>Dibble, Mack, Mehresh, and Chen, UCB</td>
<td>20 Homework Problems in Combustion using MATLAB Cantera</td>
</tr>
<tr>
<td>10:40</td>
<td></td>
<td>Break</td>
</tr>
<tr>
<td>11:00</td>
<td>D. Goodwin</td>
<td>One-Dimensional Flame Simulations</td>
</tr>
<tr>
<td>11:30</td>
<td>C. Pantano Caltech</td>
<td>Using Cantera to Construct Flamelet Libraries for Large-Eddy Simulation</td>
</tr>
<tr>
<td>12:00</td>
<td></td>
<td>Lunch</td>
</tr>
<tr>
<td>1:00</td>
<td>D, Goodwin</td>
<td>Cantera C++ Programming</td>
</tr>
<tr>
<td>1:20</td>
<td>E. D. Huckaby, DOE/NETL</td>
<td>Extending Cantera</td>
</tr>
<tr>
<td>1:40</td>
<td>E. D. Huckaby, DOE/NETL</td>
<td>Coupling Cantera to Fluent</td>
</tr>
<tr>
<td>2:00</td>
<td></td>
<td>Discussion / Future directions</td>
</tr>
</tbody>
</table>
Introduction to Cantera

David G. Goodwin

Division of Engineering and Applied Science
California Institute of Technology
Cantera is a set of software tools for problems involving kinetics

Some Capabilities:

- Thermodynamic properties
- Transport properties
- Chemical equilibrium
- Homogeneous and heterogeneous chemistry
- Reactor networks
- One-dimensional flames
- Electrochemistry
- Reaction path diagrams

And Some Features:

- MATLAB Toolbox
- Python Interface
- C++ kernel
- Fast algorithms
- Object-oriented
- Backward-compatible with Chemkin-II
- Run under Windows, Linux, Mac OS X, Unix, ...
- open source
A few things Cantera can do...
Counterflow Diffusion Flame

Non-premixed, ethane/air opposed-jet flame at 1 atm pressure
Heterogeneous chemistry

Oxidation of a1% methane in air mixture on a 1 cm$^2$ Pt foil. Pt catalytic combustion mechanism is from Deutschman et al., Proc. 26th Symp. (Intl.) on Combustion, 1996, pp. 1747-1754

![Graphs showing temperature, pressure, and coverages over time]
Transport Properties

Prandtl number of an equilibrium H/O mixture
Chemical Equilibrium

Adiabatic flame temperature for methane/air as a function of equivalence ratio
Reaction Path Diagrams

view chemical pathways at any point in a flame, or at any time in a kinetics simulation
Cantera can be used from applications written in Fortran, C, C++, ...

- Use it to
  - evaluate thermodynamic and/or transport properties
  - compute chemical equilibrium
  - calculate homogeneous and heterogeneous reaction rates
  - evaluate liquid/vapor equilibria
  - ...

- Use it with commercial CFD packages (Fluent, ...)
Example: a 2D model of a single-chamber fuel cell

- Fortran 90 main program
- Cantera used for
  - Transport properties
  - Heterogeneous reaction rates in porous electrodes
  - Multi-domain stiff solver
Another example: a transient model of SOFC "pattern anode" experiments

- C++ application
- Cantera used for
  - thermodynamic properties
  - heterogeneous reaction rates on two 2D surface phases and at the 1D "triple-phase boundary"
Validation and Benchmarks
Cantera Workshop

Benchmark Validation

SENKIN / Chemkin-II vs. SENKIN / Cantera

Cantera and Chemkin-II versions are in excellent agreement for all solution variables and sensitivity coefficients.

Typical relative difference in any component at any time is one part in $10^6$ or $10^7$. 

$max \text{ temperature error} = 0.014 \text{ K}$

$10^4(T_{\text{cantera}} - T_{\text{chemkin}})$
Benchmark Performance

Constant P, H Problem with Sensitivity Analysis
Cantera is significantly faster for this benchmark on all platforms tested
(Linux, Windows, OSF/1, IRIX)

Note: these tests were conducted with an earlier version of Cantera.
Cantera Structure
Cantera Structure

User Interface Layer
- C++ Interface (header files, cxxlib)
- Cantera Python Package (classes, functions, constants)
- MATLAB Toolbox m-files, classes

API Layer
- Python Extension Module
- MATLAB MEX file

C Layer
- clib library
- C-callable functions

Kernel
- Cantera Kernel
  - C++ Classes, Functions, Constants

External C / F77 Numerical Libraries
  - CVODE, BLAS, LAPACK, DASPK...

"Cantera"
The Cantera MATLAB Toolbox

- MATLAB is a widely-used technical problem solving environment
- Excellent capabilities for numerical mathematics
  - Linear algebra
  - ODEs, PDEs
- Toolboxes implement application-specific capabilities
- The Cantera toolbox brings kinetics, thermo, and transport functionality to MATLAB

**Example:** Prandtl number for an equilibrium O/H mixture, computed using multicomponent transport properties
The Cantera Python Interface

- Python is an easy-to-use object-oriented scripting language
- Easy to create graphical applications
- Python comes preinstalled on most Linux and Mac machines; a version for Windows is also available
- Used in several large-scale scientific computing projects for top-level user environment

Screen shot of MixMaster, a Cantera application written in Python
Cantera has a similar ‘look and feel’ in all environments

**MATLAB**

```matlab
gas = GRI30;  
set(gas,'T',300.0,'P',OneAtm,'X','CH4:1,O2:2,N2:7.52');  
equilibrat(gas,'HP');  
disp(gas)
```

**Python**

```python
from Cantera import *  
gas = GRI30()  
gas.set(T = 300.0, P = OneAtm, X = 'CH4:1,O2:2,N2:7.52')  
gas.equilibrate('HP')  
print gas
```

**C++**

```cpp
#include "Cantera.h"  
#include "GRI30.h"  
#include "equilibrium.h"

main() {
    GRI30 gas;  
    gas.setState_TPX(300.0, OneAtm, "CH4:1,O2:2,N2:7.52");  
equilibrat(gas, "HP");  
    cout << gas;
}
```
All three of these programs carry out the same
calculation, with the same results

<table>
<thead>
<tr>
<th>mole fraction</th>
<th>value</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CH2OH</td>
<td>4.114099e-17</td>
<td>4.654961e-17</td>
</tr>
<tr>
<td>CH3O</td>
<td>6.575072e-19</td>
<td>7.439468e-19</td>
</tr>
<tr>
<td>CH3OH</td>
<td>3.492911e-18</td>
<td>4.080466e-18</td>
</tr>
<tr>
<td>C2H</td>
<td>3.671127e-24</td>
<td>3.350115e-24</td>
</tr>
<tr>
<td>C2H2</td>
<td>9.146744e-22</td>
<td>8.630585e-22</td>
</tr>
<tr>
<td>C2H4</td>
<td>7.230037e-27</td>
<td>7.394897e-27</td>
</tr>
<tr>
<td>C2H5</td>
<td>7.843632e-32</td>
<td>8.310722e-32</td>
</tr>
<tr>
<td>C2H6</td>
<td>5.697162e-33</td>
<td>6.245789e-33</td>
</tr>
<tr>
<td>HCCO</td>
<td>5.366464e-20</td>
<td>8.027556e-20</td>
</tr>
<tr>
<td>CH2CO</td>
<td>7.105980e-20</td>
<td>1.089078e-19</td>
</tr>
<tr>
<td>HCCOH</td>
<td>6.968486e-23</td>
<td>1.068006e-22</td>
</tr>
<tr>
<td>N</td>
<td>1.424539e-08</td>
<td>7.274647e-09</td>
</tr>
<tr>
<td>NH</td>
<td>2.359734e-09</td>
<td>1.291754e-09</td>
</tr>
<tr>
<td>NH2</td>
<td>9.409152e-10</td>
<td>5.496479e-10</td>
</tr>
<tr>
<td>NH3</td>
<td>2.686508e-09</td>
<td>1.668083e-09</td>
</tr>
<tr>
<td>NNH</td>
<td>7.558466e-10</td>
<td>7.994703e-10</td>
</tr>
<tr>
<td>NO</td>
<td>1.882060e-03</td>
<td>2.065666e-03</td>
</tr>
<tr>
<td>NO2</td>
<td>3.462822e-07</td>
<td>5.808190e-07</td>
</tr>
<tr>
<td>N2O</td>
<td>1.001874e-07</td>
<td>1.607657e-07</td>
</tr>
<tr>
<td>HNO</td>
<td>3.366350e-08</td>
<td>3.806438e-08</td>
</tr>
<tr>
<td>CN</td>
<td>6.774013e-14</td>
<td>6.425636e-14</td>
</tr>
<tr>
<td>HCN</td>
<td>1.930651e-11</td>
<td>1.902309e-11</td>
</tr>
<tr>
<td>H2CN</td>
<td>5.477443e-18</td>
<td>5.598318e-18</td>
</tr>
<tr>
<td>HCCN</td>
<td>1.316652e-21</td>
<td>1.969694e-21</td>
</tr>
<tr>
<td>HNO2</td>
<td>1.020209e-16</td>
<td>1.600336e-16</td>
</tr>
<tr>
<td>HOCN</td>
<td>1.131156e-12</td>
<td>1.774372e-12</td>
</tr>
<tr>
<td>HNO2</td>
<td>3.864622e-10</td>
<td>6.062186e-10</td>
</tr>
<tr>
<td>NO2</td>
<td>1.543840e-11</td>
<td>2.364990e-11</td>
</tr>
<tr>
<td>N2</td>
<td>7.085838e-01</td>
<td>7.237005e-01</td>
</tr>
<tr>
<td>AR</td>
<td>0.000000e+00</td>
<td>0.000000e+00</td>
</tr>
<tr>
<td>C3H7</td>
<td>3.540964e-47</td>
<td>5.562683e-47</td>
</tr>
<tr>
<td>C3H8</td>
<td>2.428377e-48</td>
<td>3.904100e-48</td>
</tr>
<tr>
<td>CH2CHO</td>
<td>2.952869e-25</td>
<td>4.634145e-25</td>
</tr>
<tr>
<td>CH3CHO</td>
<td>5.854760e-26</td>
<td>9.403438e-26</td>
</tr>
</tbody>
</table>
The Cantera Kernel

- C++ class libraries

- Designed for performance
  - Property caching
  - Virtual methods used sparingly
  - Low-level methods written to maximize inlining
  - CPU-intensive code hand-optimized

- Uses standard open-source numerical libraries
  - BLAS, LAPACK, CVODE, DASPK
What's on the CD?

- **Documentation**
  - HTML pages documenting
    - The Cantera Python Package
    - The C++ kernel
  - Documents and presentations describing input files, F77 interfacing, installation procedures, etc.
  - Some workshop materials

- **Source Code**
  - The latest version of the full source distribution

- **Third-party applications**
  - Python 2.3.4, Numeric, and graphviz installers (Win/Mac)

- **Binary installers for Windows and Mac platforms**
  - MATLAB toolbox
  - Python Package
  - libraries and header files for building C++ or Fortran applications