SrmFOAM：An OpenFOAM-powered Solver for Simulating Working Process of Solid Rocket Motors

Presentation · July 2023

DOI: 10.13140/RG.2.2.26837.23528

1 author:

Li Wentao

Beihang University (BUAA)

13 PUBLICATIONS **10** CITATIONS

SEE PROFILE

北京航空航天大學
BEIHANG UNIVERSITY

SrmFOAM:
An OpenFOAM-powered Solver for Simulating Working Process of Solid Rocket Motors

Reporter: Li Wentao （李文韬）

SPACE POWER LABORATORY
SCHOOL OF ASTRONAUTICS

北京航空航天大學
BEIHANG UNIVERSITY

Brief

- **Reporter:** Li Wentao （李文韬）
- **Supervisor:** Prof. Liang Guozhu （梁国柱），Dr. He Yunqin （何允钦）
- **Institution:** School of Astronautics，Beihang University
- **Research direction:**

  Optimized design and simulation of solid rocket motors

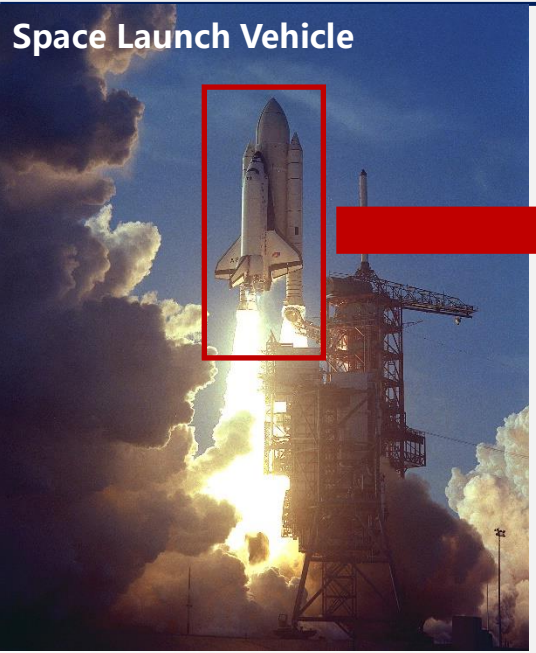
北京航空航天大學
BEIHANG UNIVERSITY

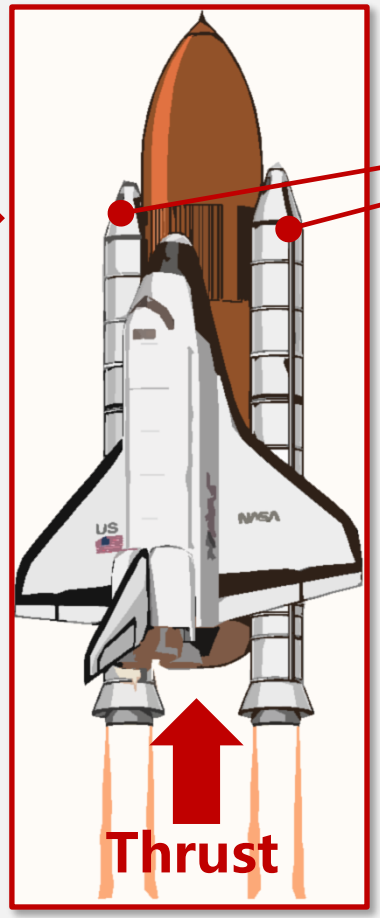
Content

北京航空航天大學
BEIHANG UNIVERSITY

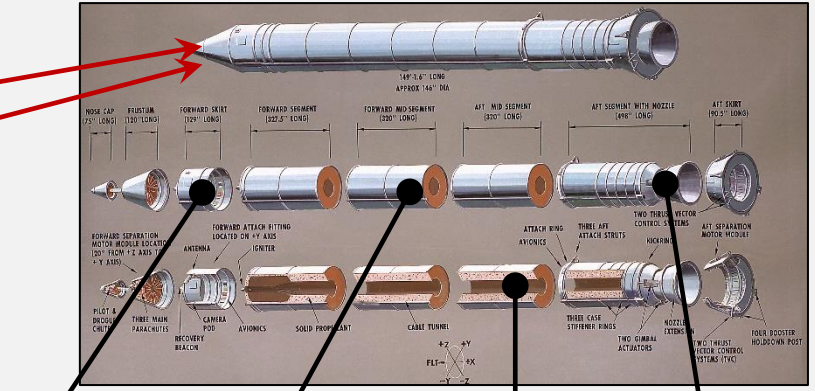
Content

Introduction





Solid Rocket Motor

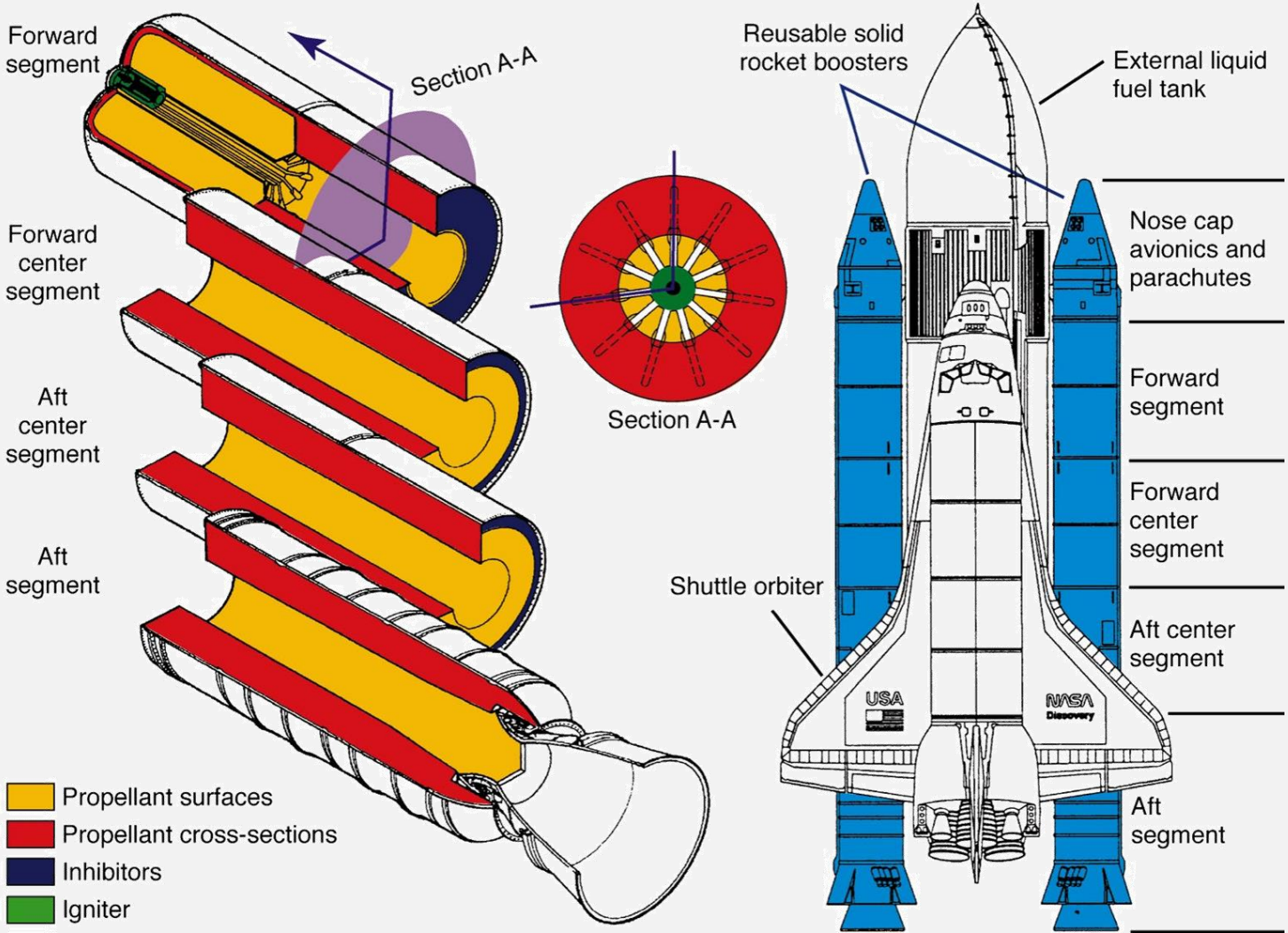


Burning Area → Chamber Pressure → Thrust

Space Shuttle

Thrust 1400 tons

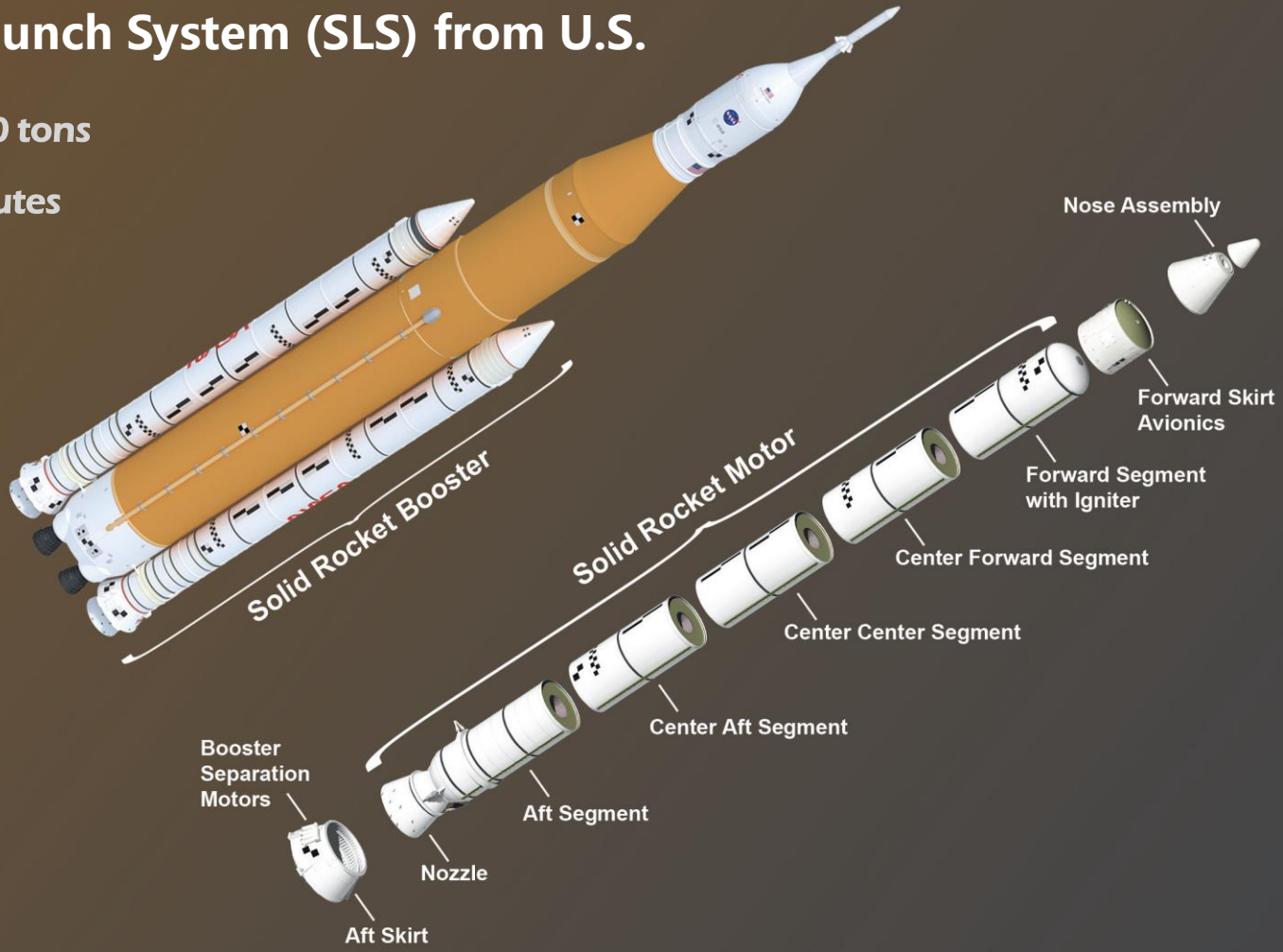
Time 2 minutes



Space Launch System (SLS) from U.S.

Thrust 1600 tons

Time 2 minutes



ARTEMIS

➢ Solid rocket motor in Europe

- Europe's Ariane 5 rocket is famous for launching the James Webb Space Telescope
- It has completed its final flight in this year

- Europe's most powerful rocket, Ariane 6, is about to launch for testing.

Kuaizhou 11 (快舟11)
Commercial Launch Vehicle

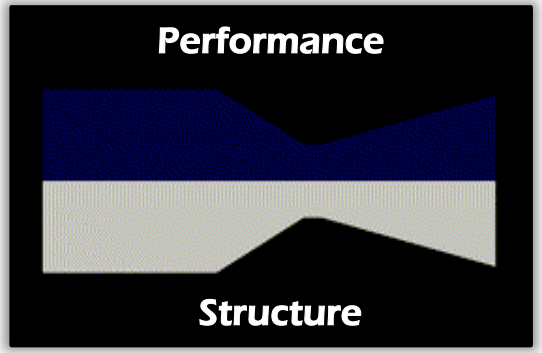
Long March 6A (长征6号甲)

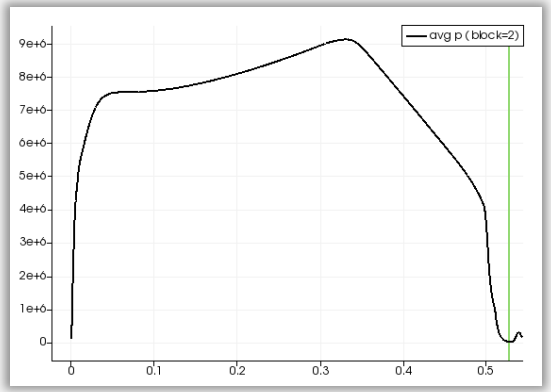
➢ Solid rocket motor in China

- Kuaizhou 11 is built with carbon fiber composite material, and its overall technical level has reached the state-of-the-art level.
- Long March 6A is the first Chinese launch vehicle boosted by solid rockets.
- Advanced 500-tons-thrust motor is the most powerful motor (without segmented propellant) in the world.

Advanced motor
with a thrust of 500 tons.

Internal ballistics





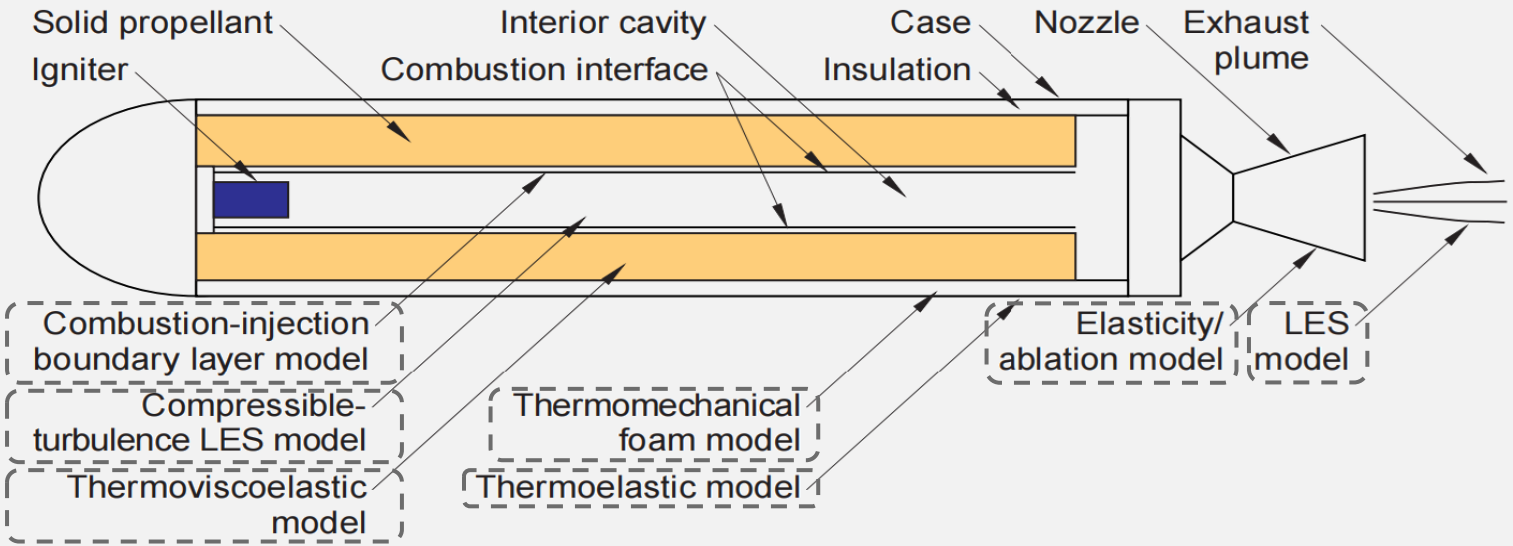
➢ Mass conservation law

$$\dot{m}_p = \dot{m}_e + \dot{m}_s$$

$$\begin{cases} \dfrac{\mathrm{d}p_c}{\mathrm{d}t} = \dfrac{\left(c^*\Gamma\right)^2}{V_f}\left(a\rho_p A_b\left(w\right)p_c^n - \dfrac{p_c A_t}{c^*}\right) - \dfrac{A_b a p_c^{n+1}}{V_f} \\ \dfrac{\mathrm{d}V_f}{\mathrm{d}t} = A_b a p_c^n \\ \dfrac{\mathrm{d}w}{\mathrm{d}t} = a p_c^n \end{cases}$$

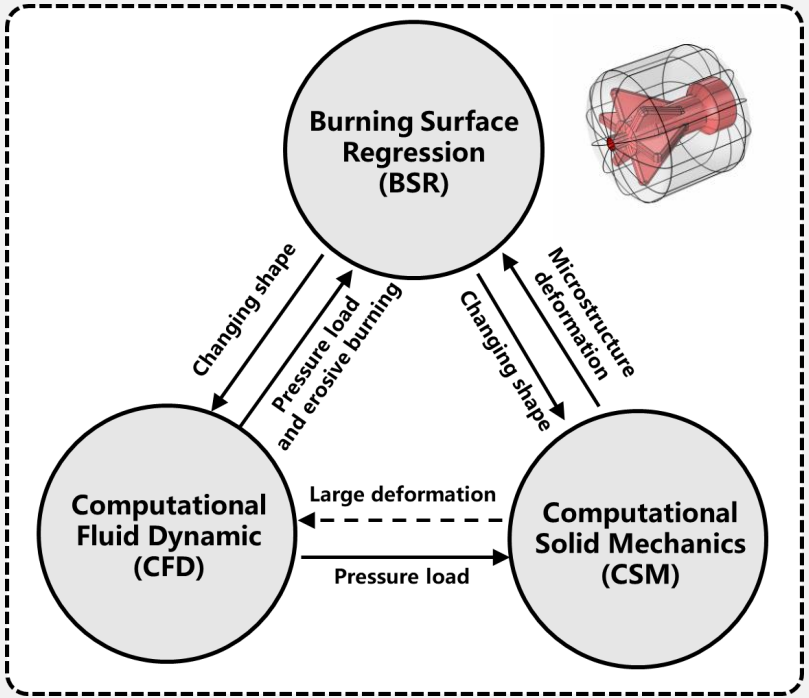
However, it is much more complex in a real motor...

Physical problems



- ✓ (1) Moving-boundary enthalpy-adding multiphase chemical reaction turbulent flow
- ✓ (2) Complex shock wave system at the outlet of the nozzle
- ✓ (3) Structural response of the solid grain and other nonlinear anisotropic materials
- ✓ (4) Combustion of composite propellant and its burning surface regression
- ✓ (5) Ablation of the nozzle throat
- ✓ (6) Ignition mechanism and flameout process

Physical problems



➢ Critical Multiphysics problem

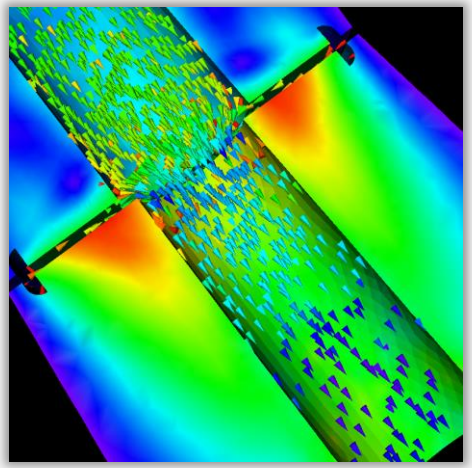
- The BSR can change the shape of the fluid and solid regions, and on the other hand the CFD and CSM results determine the solid propellant's regression rate (or burning rate).
- The CFD and CSM also form a fluid-structure interaction (FSI) problem.
- The combustion process can be modeled by the injection of burned gas
- Since the deformation of the grain is not large, the effect of deformation on the flow field is usually not considered.

➢ ROCSTAR Multiphysics ——The pioneer of SRM Multiphysics simulation

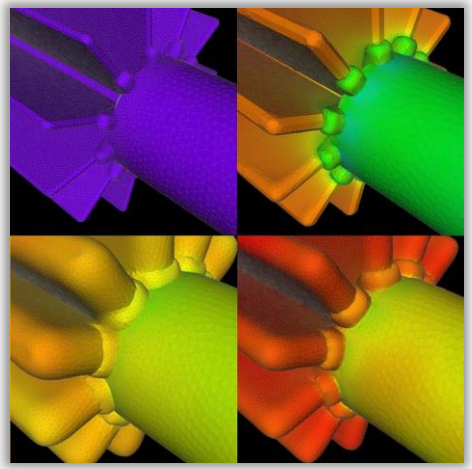
- The ROCSTAR Multiphysics is an open-source software that has been under continuous development for almost two decades.
- The software uses two mesh sets for fluid (using finite volume method, FVM) and solid (using finite element method, FEM), requiring an accurate mesh-to-mesh data transfer at the gas-solid interface.
- The BSR is realized by mesh motion and smoothing technique.

ILLINOIS ROCSTAR

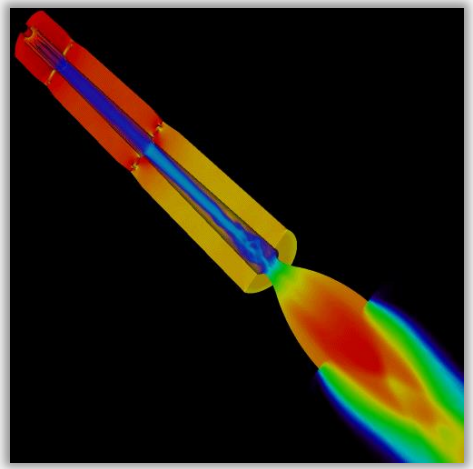
Logo



Grain deformation



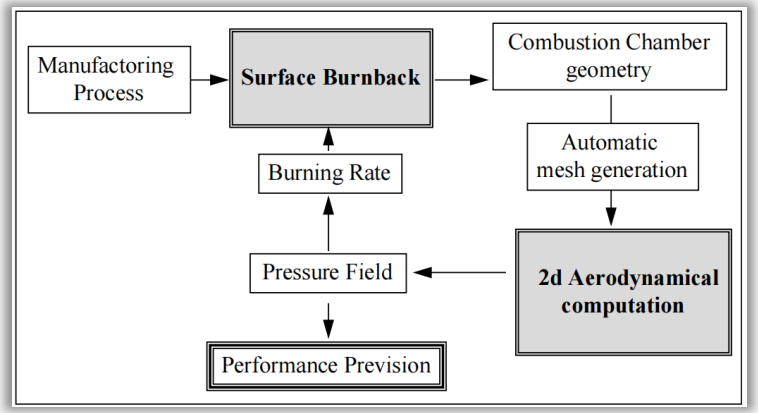
Surface regression



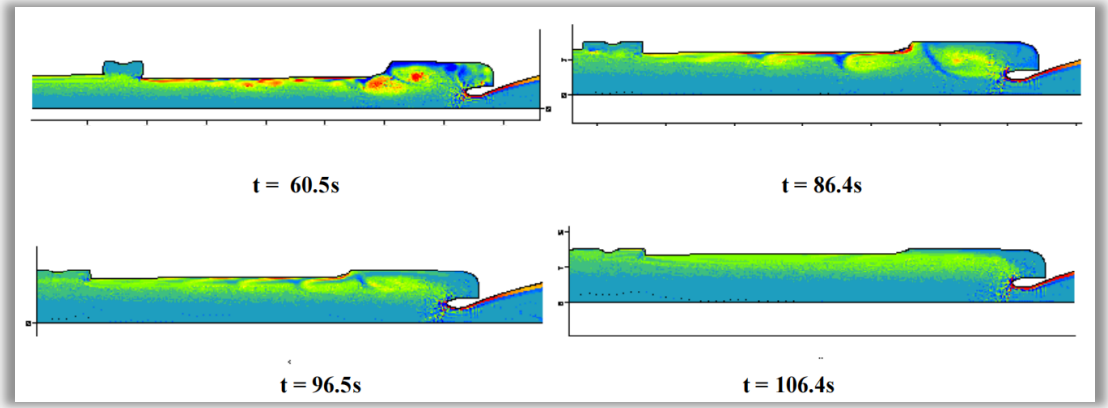
Whole field

➢ MOPTI——An European Software

- MOPTI is a computational code developed in France since 1998 for simulating the performance of SRMs, but it does not include solid mechanics.
- Similar to the ROCSTAR Multiphysics, MOPTI manages exchanges between two primary modules, that is, burning surface regression (BSR) and computational fluid dynamics (CFD).
- MOPTI can provide valuable insights into the behavior of solid rocket and help identify potential issues, such as thrust oscillations.



Framework



Vorticity fields

北京航空航天大學
BEIHANG UNIVERSITY

Content

Expectations

➢ Dynamic mesh ⟶ Static mesh

To seek an alternative approach, where the capturing of the burning surface and simulation of the flow field can be achieved under a static mesh instead of a dynamic mesh.

➢ Finish computational solid mechanics (CSM) on this static mesh

The solid boundary of grain changes as the burning surface moves, making it challenging to add pressure load using a static mesh.

➢ Solver is lightweight and easy to use

The software, such as ROCSTAR Multiphysics, has dozens of dependencies. It makes installation extremely difficult for users. We expect a lightweight framework with standard user interface enabling "out-of-the-box" use.

Objective

★ Develop a lightweight multi-physics solver for solid rocket motor simulation using OpenFOAM

SrmFOAM
SOLID ROCKET MOTOR

Features

OpenFOAM
The OpenFOAM Foundation

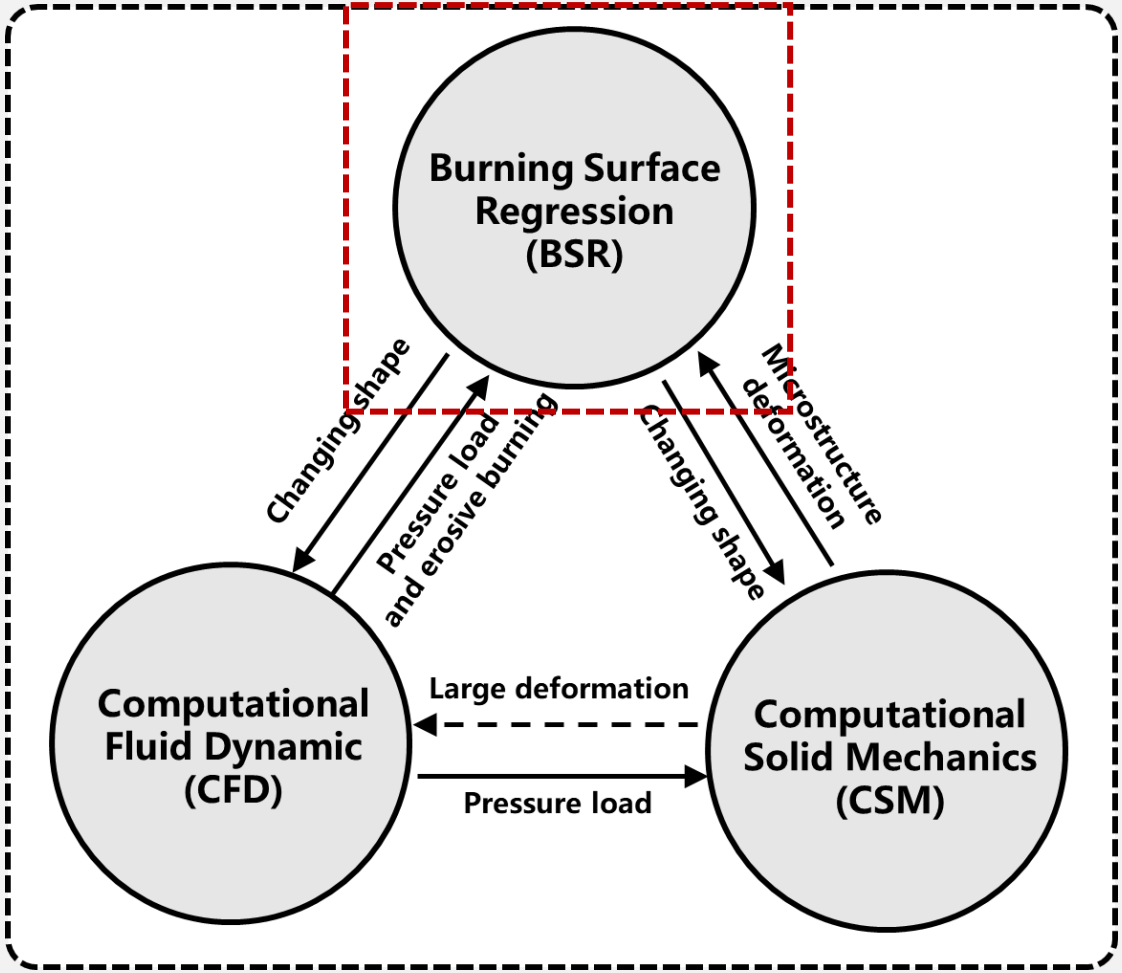
(1) It can simulate the full burnout of SRMs in massively parallel environments.

(2) BSR, CFD, and CSM are calculated using the finite volume method on a static unstructured mesh.

(3) The linearized eikonal equation is employed for burning surface regression and capture.

(4) The solver can consider erosive burning phenomena and the effect of grain strain on the burning rate.

(5) No third-party dependencies are required aside from OpenFOAM, enabling "out-of-the-box" use with a single executable file for the entire simulation.


Burning Surface Regression (BSR)
Changing shape
Pressure load and erosive burning
Changing shape
Microstructure deformation
Computational Fluid Dynamic (CFD)
Large deformation
Pressure load
Computational Solid Mechanics (CSM)

➢ The definition of burned time field

Burned time field represents the moment that the burning surface crosses the location

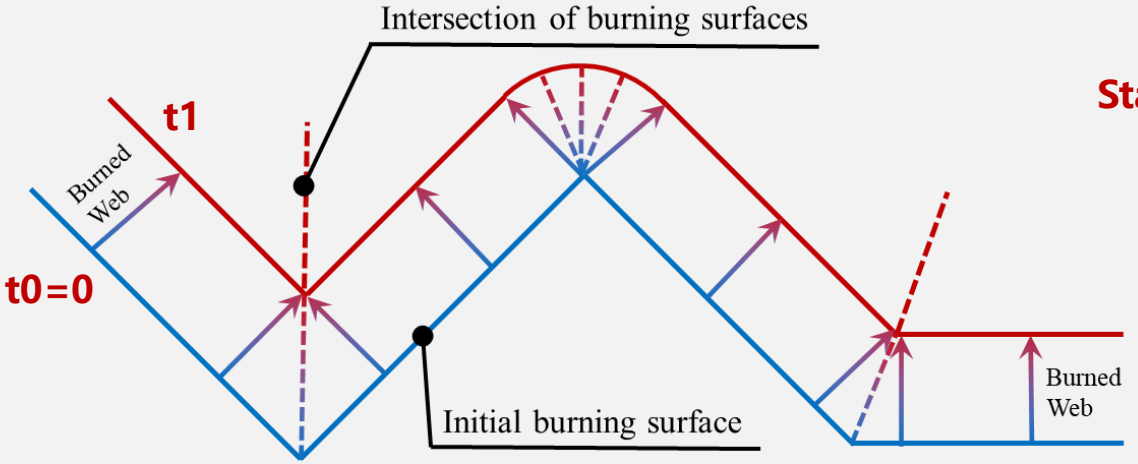
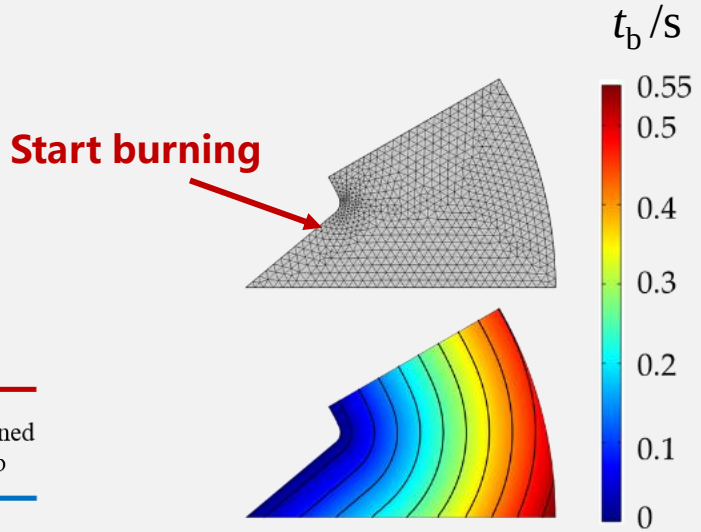


Illustration of BSR



Burned time field of a star grain

The burning surface at any moment can be captured by the iso-surfaces of the burned time field

Burning surface regression

How to solve the burned time field?

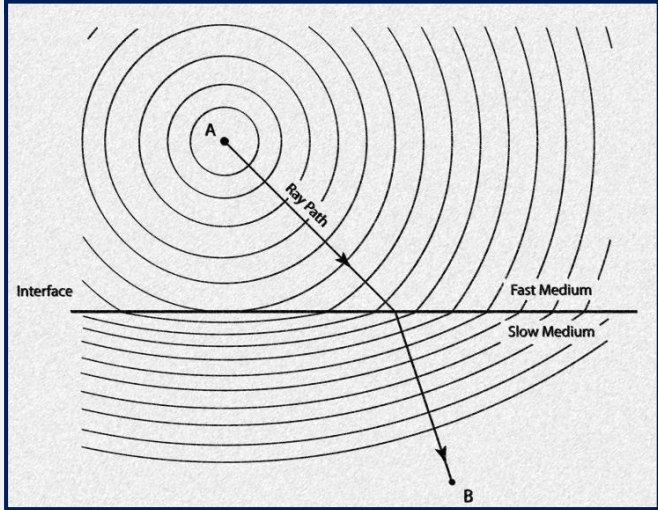
The eikonal equation

$$\begin{cases} r_b^{\,2}\nabla t_b \cdot \nabla t_b = 1, & \eta_0 = 1 \quad \text{At grain zone} \\ t_b = 0, & \eta_0 = 0 \quad \text{At gas zone} \end{cases}$$

How to solve the eikonal equation?

- **Hyperbolic method:**
  Fast sweep method
- **Elliptical method:**
  Artificial diffusion

$$\begin{cases} \nabla \cdot \left(\varepsilon_r r_b h \nabla t_b \right) = r_b^{\,2}\nabla t_b \cdot \nabla t_b - 1, & \eta_0 = 1 \\ t_b = 0, & \eta_0 = 0 \end{cases}$$

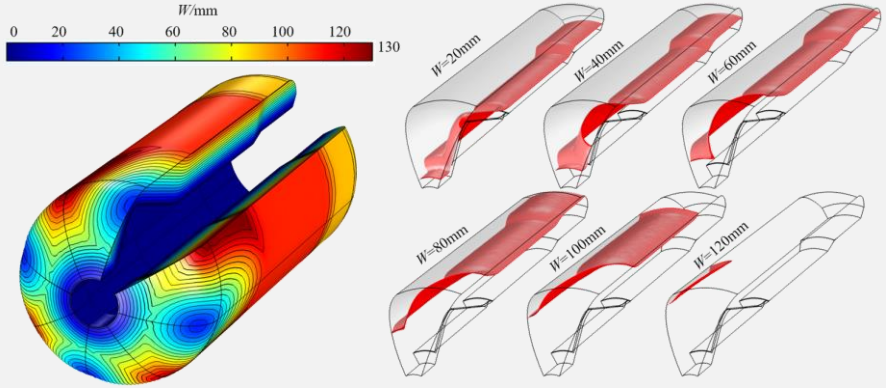


Burning surface regression

With out linearization

$$\begin{cases} \nabla\cdot\left(\varepsilon_r r_b h\nabla t_b\right) = r_b{}^2\nabla t_b\cdot\nabla t_b - 1, \quad \eta_0 = 1 \\ t_b = 0, \quad \eta_0 = 0 \end{cases}$$

- The equation can be discretized into a large quadratic equation system
- The results can be solved using the Newtonian iterative method
- It is suitable for COMSOL but not OpenFoam



Linearization method

$$\begin{cases} \nabla\cdot\left(\varepsilon_r r_b h\nabla t_b\right) - \dfrac{t_b}{\tau} = r_b{}^2\nabla t_b{}^{old}\cdot\nabla t_b{}^{old} - 1 - \dfrac{t_b{}^{old}}{\tau}, \quad \eta_0 = 1 \\ t_b = 0, \quad \eta_0 = 0 \end{cases}$$

$$\tau = 0.2\frac{h}{r_b}$$

- "Old" refers to the result obtained from the previous step in the iterative process
- Red terms ensure that the coefficient matrix obtained by discretization is diagonally dominant to stabilize the iteration.

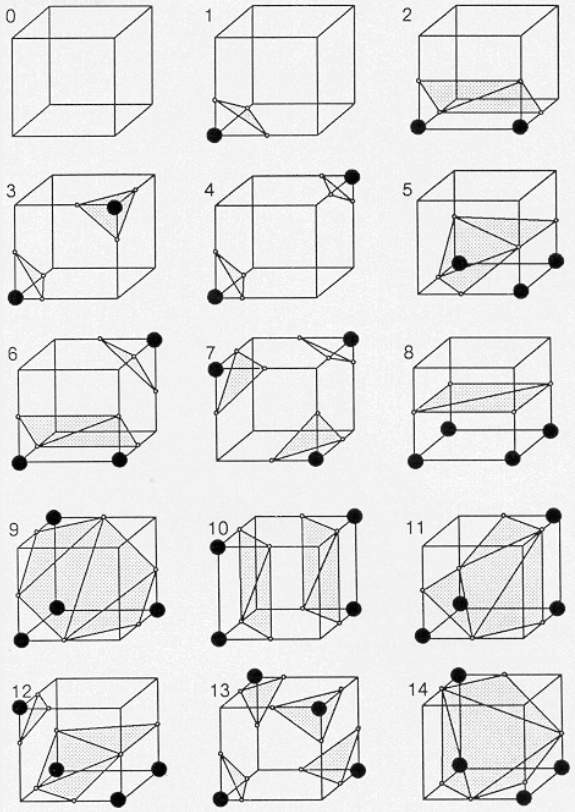
A compact form

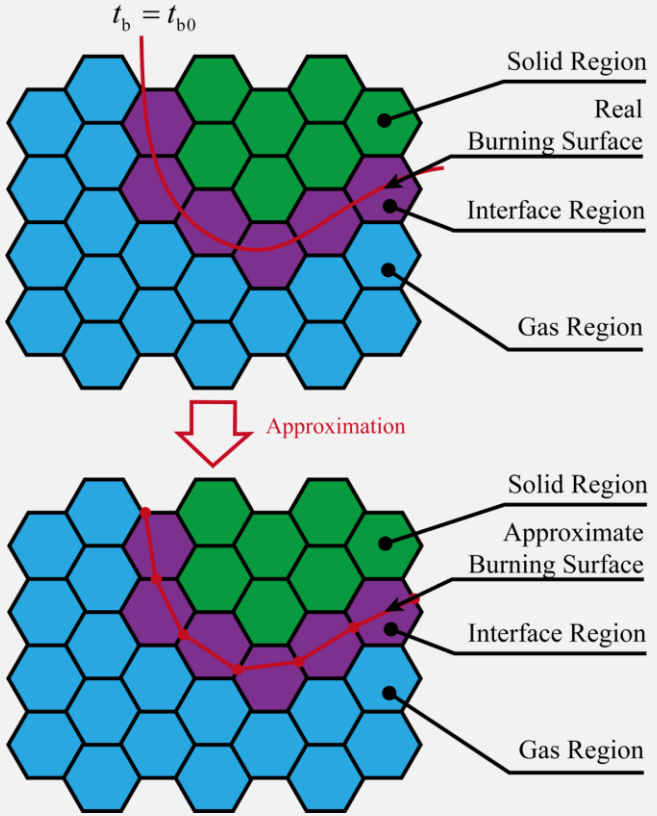
$$\nabla\cdot\left(\varepsilon_r\eta_0 r_b h\nabla t_b\right) - \frac{t_b}{\tau} = \eta_0\left(r_b{}^2\nabla t_b{}^{old}\cdot\nabla t_b{}^{old} - 1 - \frac{t_b{}^{old}}{\tau}\right)$$

Remember: It is an iteration process!

Burning surface capture

- The marching cubes algorithm (MCA), which was introduced by Lorensen and Cline in 1987, is one of the primary methods used for generating or extracting iso-surfaces from a given field.
- The isoSurface class in OpenFOAM uses a modified version of marching cubes algorithm, enabling it to work with the finite volume method and unstructured meshes.



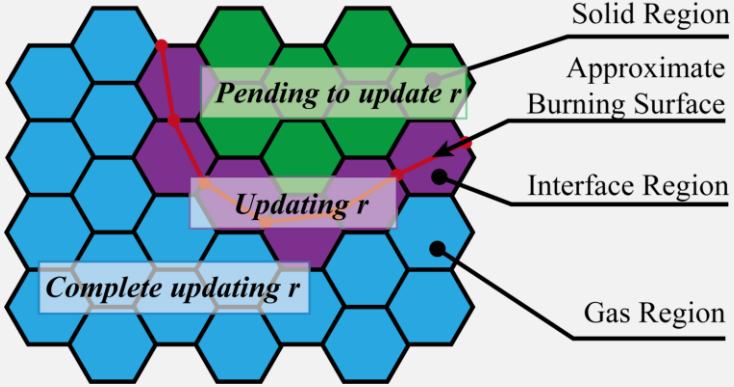


Burning surface regression

One remaining problem

$$\nabla\cdot\left(\varepsilon_{\mathrm{r}}\eta_0 r_{\mathrm{b}} h\nabla t_{\mathrm{b}}\right)-\frac{t_{\mathrm{b}}}{\tau}=\eta_0\left(r_{\mathrm{b}}{}^2\nabla t_{\mathrm{b}}{}^{\mathrm{old}}\cdot\nabla t_{\mathrm{b}}{}^{\mathrm{old}}-1-\frac{t_{\mathrm{b}}{}^{\mathrm{old}}}{\tau}\right)$$

Only the cells swept by interface region have accurate burning rate, while the cells not swept by interface region have undefined burning rate



Burning Rate Extension Method

$$\begin{cases} r_{\mathrm{b}}=r_{\mathrm{b}}^{\mathrm{old}}, \quad \eta_{\mathrm{rb}}=1 \\ \nabla\cdot\left(h^2\nabla r_{\mathrm{b}}\right)=0, \quad \eta_{\mathrm{rb}}=0 \end{cases}$$

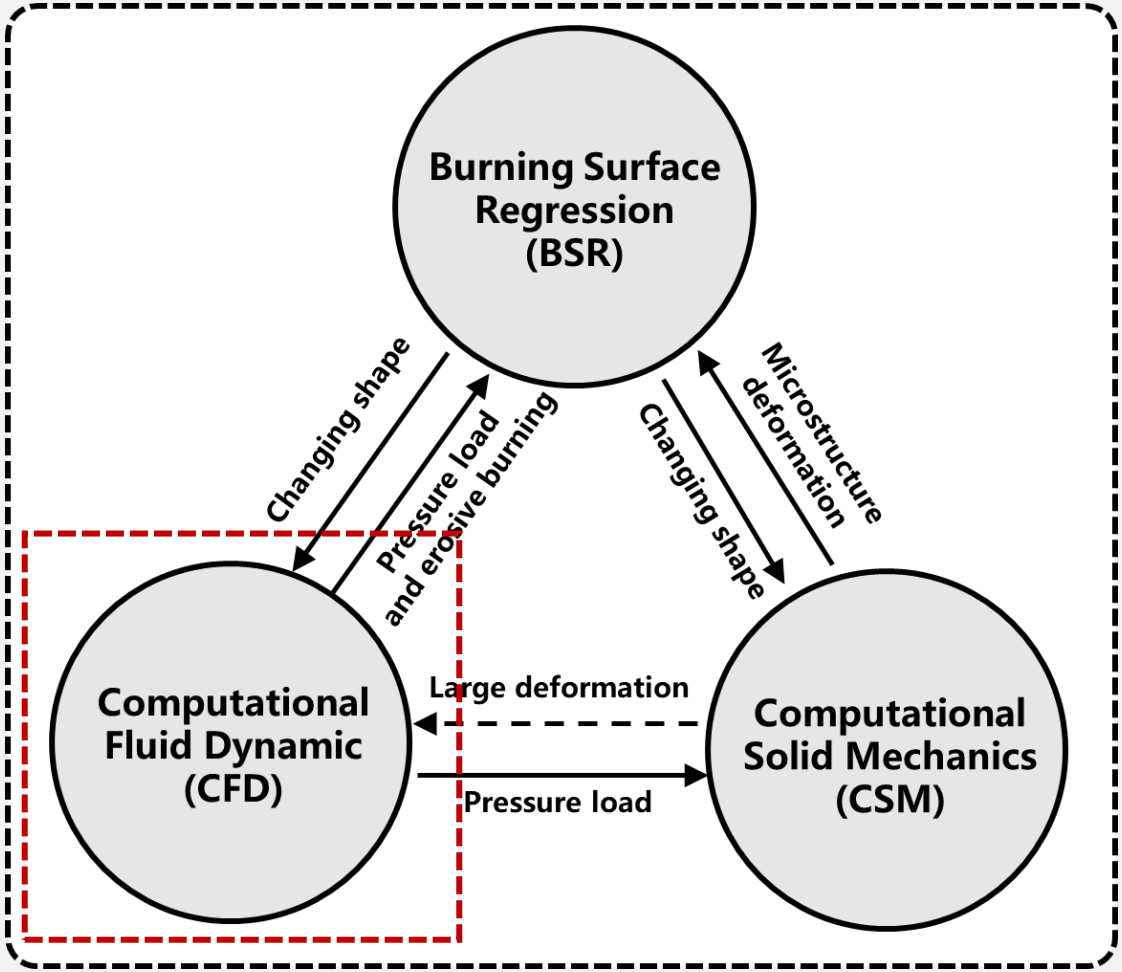
→ Where the local burning rate is known

→ Where the burning rate is unknown and need to be extended

OR

$$\nabla\cdot\left(\left(1-\eta_{\mathrm{rb}}\right)h^2\nabla r_{\mathrm{b}}\right)=\eta_{\mathrm{rb}}\left(r_{\mathrm{b}}-r_{\mathrm{b}}^{\mathrm{old}}\right)$$

The equation can extend the burning rate to the entire field


Burning Surface Regression (BSR)
Changing shape
Pressure load and erosive burning
Changing shape
Microstructure deformation
Computational Fluid Dynamic (CFD)
Large deformation
Pressure load
Computational Solid Mechanics (CSM)

Navier-Stokes equations

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{v}) = S_{\mathrm{m}} \\ \dfrac{\partial (\rho \boldsymbol{v})}{\partial t} + \nabla \cdot (\rho \boldsymbol{v}\boldsymbol{v}) = -(1-\eta_{\mathrm{s}})\nabla p + \nabla \cdot \left(\mu^{\mathrm{e}} \nabla \boldsymbol{v}\right) + \dfrac{1}{3}\nabla \left(\mu^{\mathrm{e}} \nabla \cdot \boldsymbol{v}\right) + S_v \\ \dfrac{\partial (\rho h^*)}{\partial t} + \nabla \cdot (\rho \boldsymbol{v} h^*) - \dfrac{\mathrm{d}p}{\mathrm{d}t} = \nabla \cdot (\boldsymbol{\tau} \boldsymbol{v}) + \nabla \cdot \left(\lambda^{\mathrm{e}} \nabla T\right) + S_h \end{cases}$$

Continuity equation

Solid marker

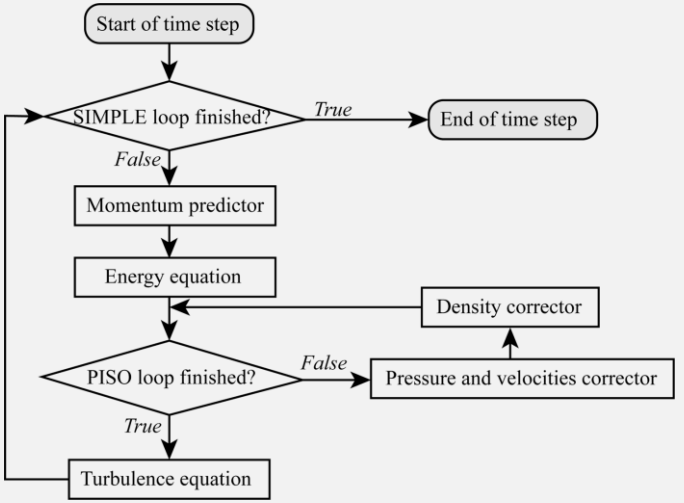
Momentum equation

Total enthalpy equation

Note: μ^{e} and λ^{e} are the equivalent dynamic viscosity and thermal conductivity, related to the choice of turbulence model and gas properties

PIMPLE Algorithm

PIMPLE algorithm is a combination of the SIMPLE and PISO algorithms. One time step contains a SIMPLE outer loop and a PISO inner loop.



Source terms

Source term of continuity equation

$$S_{\mathrm{m}} = \frac{r_{\mathrm{b}} \rho_{\mathrm{p}} \|\boldsymbol{A}\|}{V}$$

- A is a vector field, represents the area vector of the local cell burning surface.
- It simulate the mass injection at the burning surface.

Source term of momentum equation

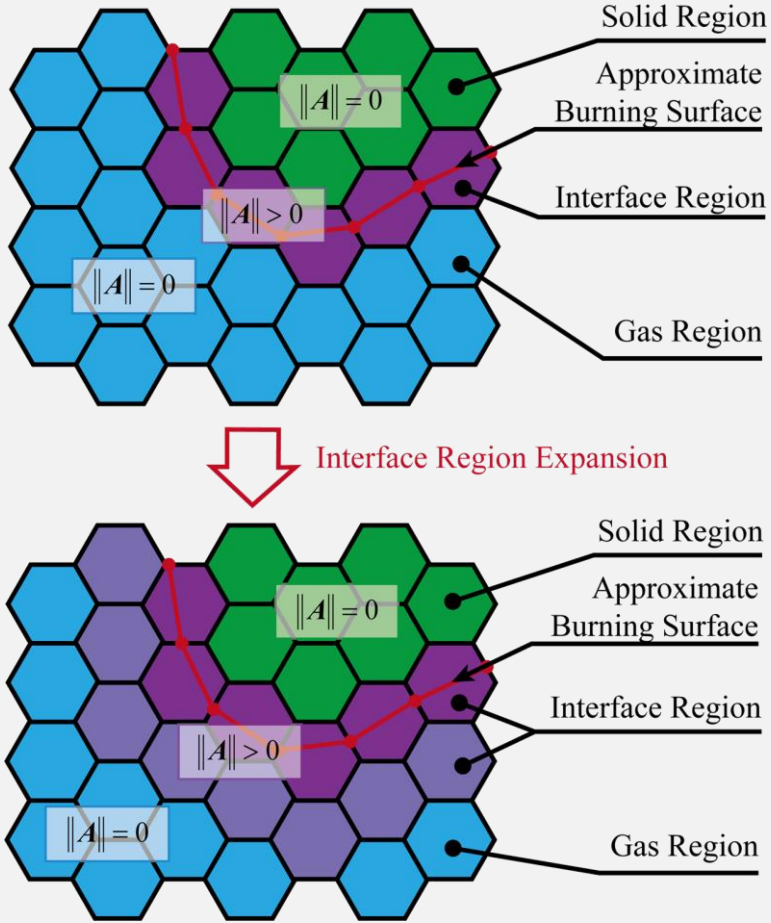
$$S_v = -\frac{\left(r_{\mathrm{b}} \rho_{\mathrm{p}}\right)^2 \boldsymbol{A}}{V \rho} - \kappa_v \eta_{\mathrm{s}} \boldsymbol{v}$$

- The first term represents the momentum carried by the injection.
- The second term represents the resistance at the solid zone.

Source term of total enthalpy equation

$$S_h = S_{\mathrm{m}} \Delta h$$

- It represents the enthalpy carried by the injection.

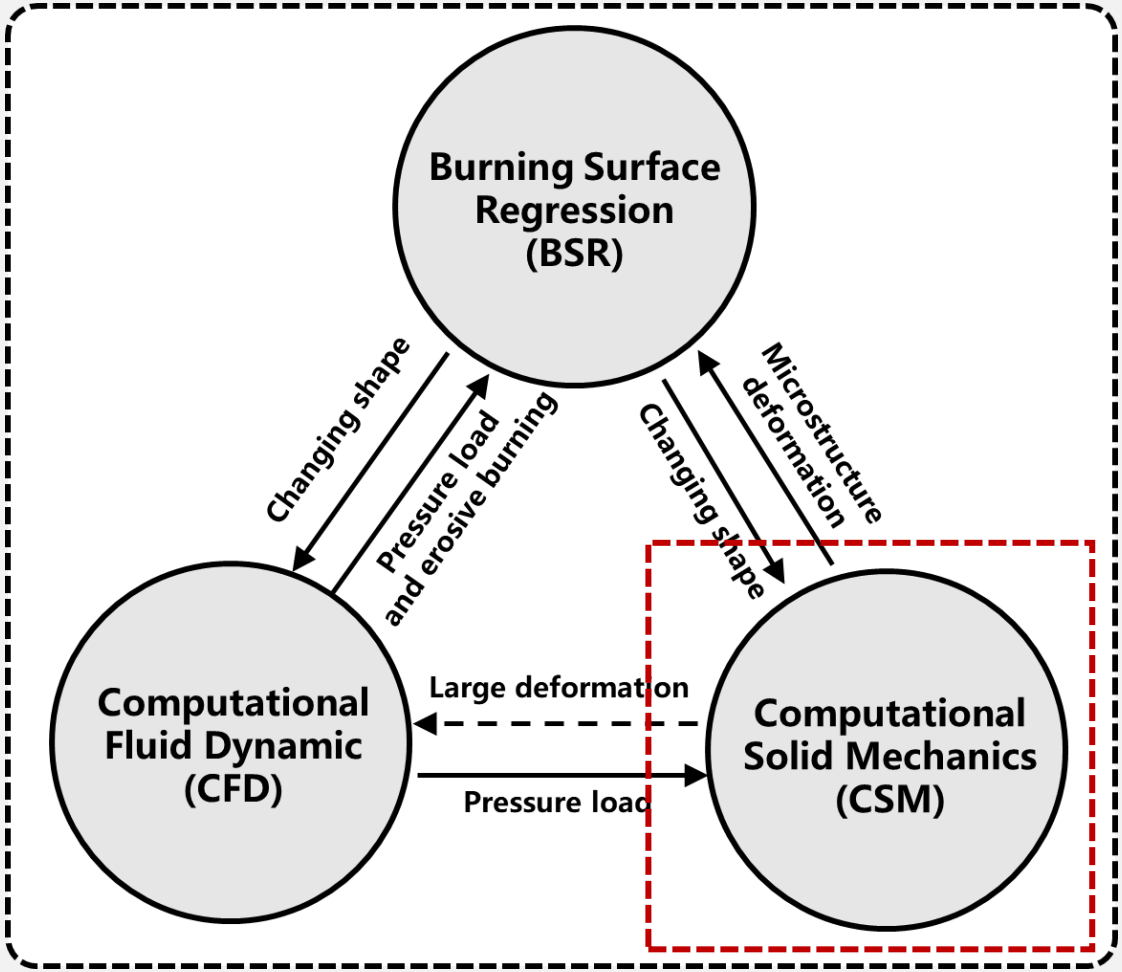


$$S_m = \frac{r_b \rho_p \|A\|}{V}$$

➢ Interface region expansion

Adding excessively large source terms into a single layer of cells may result in numerical difficulties, particularly in unstructured polyhedral meshes. Therefore, it is necessary to distribute the vector A to the surrounding cells to expand the interface region

For a cell, vector A can be shared by the nearby cells to expand the interface region without change the burning surface area significantly


Burning Surface Regression (BSR)
Changing shape
Pressure load and erosive burning
Changing shape
Microstructure deformation
Computational Fluid Dynamic (CFD)
Large deformation
Pressure load
Computational Solid Mechanics (CSM)

➢ Finite element method (FEM) or Finite volume method (FVM)?

- FEM is good at dealing with solid mechanics and FVM is good at dealing with fluid dynamics.
- However, if we use FEM, the data exchange between solid and fluid is a challenging problem.
- To avoid such difficulty, we chooses the FVM, instead of FEM.

➢ Basic equations of solid mechanics

1. Balance equation

$$\frac{\partial^2(\rho_p \boldsymbol{u}_p)}{\partial t^2} - \nabla \cdot \sigma = \boldsymbol{f}$$

Connect stress and load

2. Geometric equation

$$\varepsilon = \frac{1}{2}\left[\nabla \boldsymbol{u}_p + (\nabla \boldsymbol{u}_p)^T\right]$$

Connect strain and deformation

3. Principal equation

$$\sigma = 2\mu\varepsilon + \lambda\,\mathrm{tr}(\varepsilon)\mathbf{I}$$

$$\mu = \frac{E}{2(1+v)}$$

$$\lambda = \frac{vE}{(1+v)(1-2v)}$$

Connect stress and strain

$$\frac{\partial^2(\rho_p \boldsymbol{u}_p)}{\partial t^2} - \nabla \cdot \left[\mu\nabla \boldsymbol{u}_p + \mu(\nabla \boldsymbol{u}_p)^T + \lambda\mathbf{I}\,\mathrm{tr}(\nabla \boldsymbol{u}_p)\right] = \boldsymbol{f}$$

Control equation of deformation

Control equations

$$\frac{\partial^2(\rho_p \boldsymbol{u}_p)}{\partial t^2} - \nabla \cdot \left[\mu \nabla \boldsymbol{u}_p + \mu (\nabla \boldsymbol{u}_p)^T + \lambda \mathbf{I} \operatorname{tr}(\nabla \boldsymbol{u}_p) \right] = \boldsymbol{f}$$

To make the calculation stable, the control equations are recommended to write in the following iterative form

$$\frac{\partial^2(\rho_p \boldsymbol{u}_p)}{\partial t^2} - \nabla \cdot \left((2\mu + \lambda) \nabla \boldsymbol{u}_p \right) = \boldsymbol{f} + \nabla \cdot \left(\mu (\nabla \boldsymbol{u}_p^{\,\mathrm{old}})^T + \lambda \mathbf{I} \operatorname{tr}(\nabla \boldsymbol{u}_p^{\,\mathrm{old}}) - (\mu + \lambda) \nabla \boldsymbol{u}_p^{\,\mathrm{old}} \right)$$

Remove the non-steady state term

$$\nabla \cdot \left((2\mu + \lambda) \nabla \boldsymbol{u}_p \right) + \boldsymbol{f} + \nabla \cdot \left(\mu (\nabla \boldsymbol{u}_p^{\,\mathrm{old}})^T + \lambda \mathbf{I} \operatorname{tr}(\nabla \boldsymbol{u}_p^{\,\mathrm{old}}) - (\mu + \lambda) \nabla \boldsymbol{u}_p^{\,\mathrm{old}} \right) = 0$$

Introduce a blending factor to distinguish gas and solid

$$\theta = \begin{cases} 1, & \text{Solid and Interface Region} \\ \varepsilon_\theta, & \text{Gas Region} \end{cases}$$

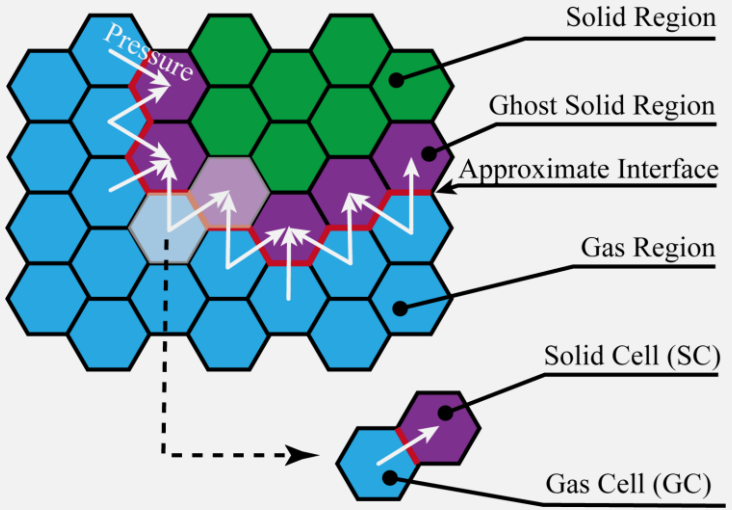
$$\nabla \cdot \left(\theta (2\mu + \lambda) \nabla \boldsymbol{u}_p \right) + \boldsymbol{f} + \nabla \cdot \left(\theta \left(\mu (\nabla \boldsymbol{u}_p^{\,\mathrm{old}})^T + \lambda \mathbf{I} \operatorname{tr}(\nabla \boldsymbol{u}_p^{\,\mathrm{old}}) - (\mu + \lambda) \nabla \boldsymbol{u}_p^{\,\mathrm{old}} \right) \right) = 0$$

Source terms

$$\nabla\cdot\left(\theta\left(2\mu+\lambda\right)\nabla\boldsymbol{u}_{\mathrm{p}}\right)+\boldsymbol{f}+\nabla\cdot\left(\theta\left(\mu(\nabla\boldsymbol{u}_{\mathrm{p}}{}^{\mathrm{old}})^{\mathrm{T}}+\lambda\mathbf{I}\,\mathrm{tr}(\nabla\boldsymbol{u}_{\mathrm{p}}{}^{\mathrm{old}})-\left(\mu+\lambda\right)\nabla\boldsymbol{u}_{\mathrm{p}}{}^{\mathrm{old}}\right)\right)=0$$

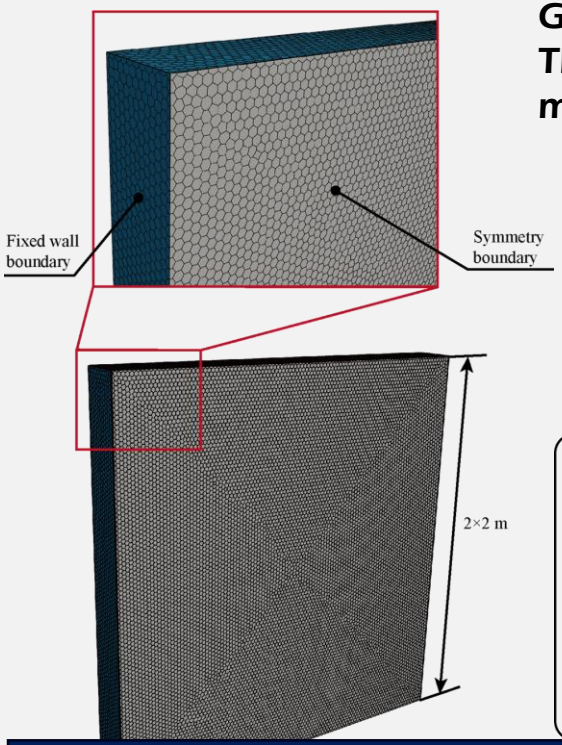
Source term ***f*** simulates gas pressure loaded on the gas-solid interface

- The mesh cell surfaces are treated as an approximate interface (AI) between the gas and solid regions
- Any individual face in the AI connects two cells, namely the gas cell (GC) and solid cell (SC)



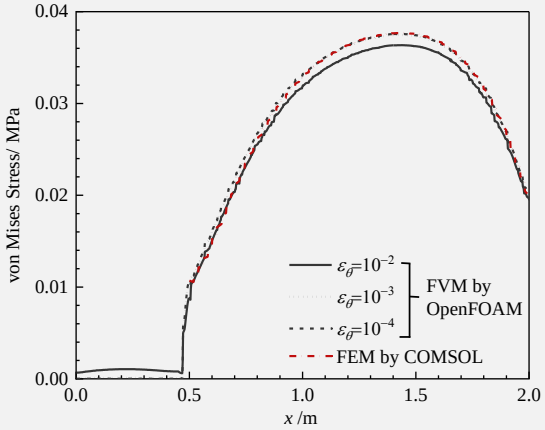
$$\boldsymbol{f}\left(x_{\mathrm{GC},i}\right)=\boldsymbol{f}\left(x_{\mathrm{SC},i}\right)=\frac{p\left(x_{\mathrm{GC},i}\right)\boldsymbol{A}_i}{V_{\mathrm{GC},i}+V_{\mathrm{GS},i}},\quad i\in\mathrm{AI}$$

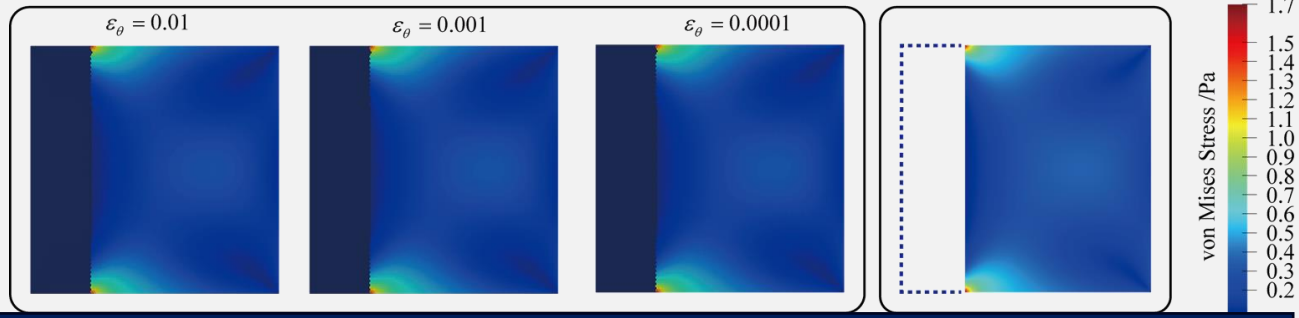
➢ Validation Case



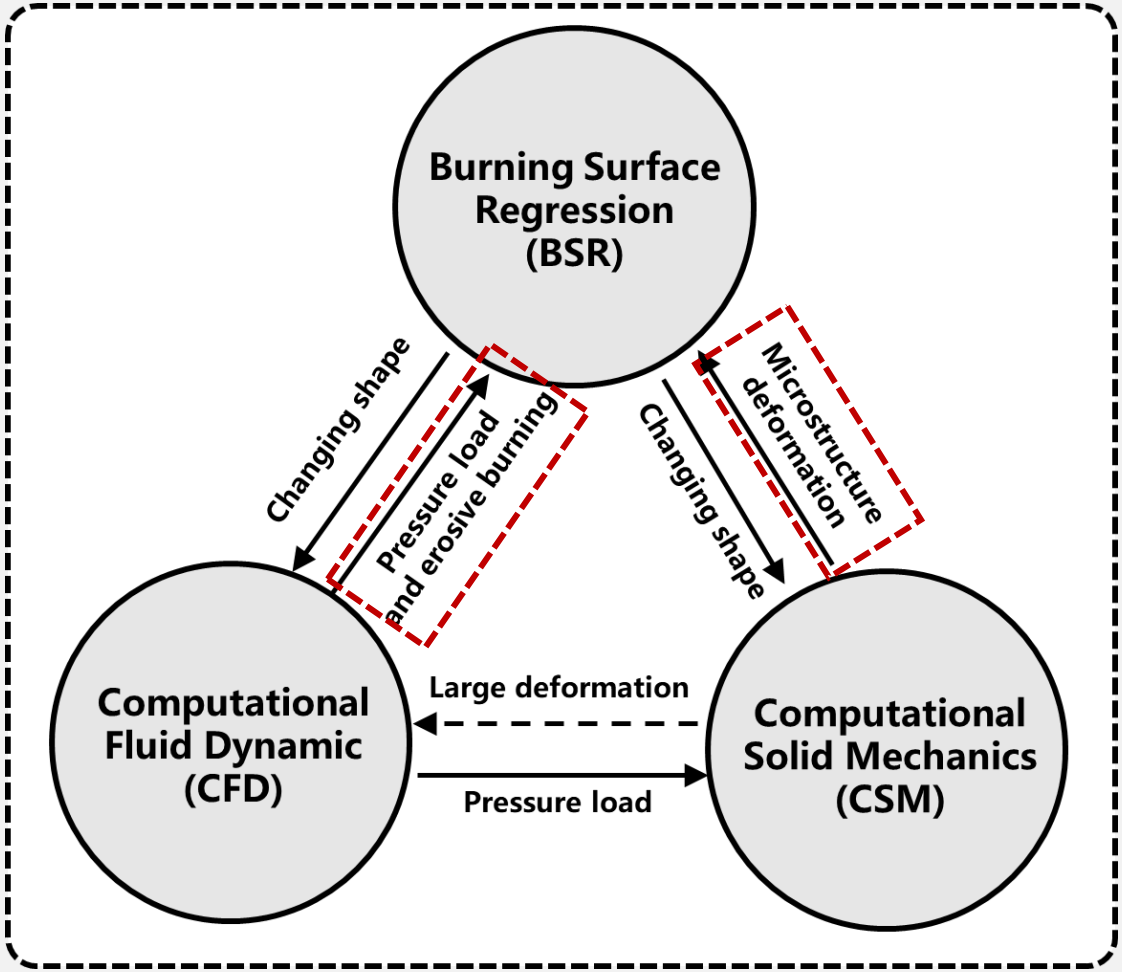
Gas (x < 0.5 m) and solid (x > 0.5). The Poisson ratio is 0.498, and the modulus of elasticity is 3.5 MPa.

$$\theta = \begin{cases} 1, & \text{Solid and Interface Region} \\ \varepsilon_\theta, & \text{Gas Region} \end{cases}$$





The results are considered sufficient for the engineering needs


Burning Surface Regression (BSR)
Changing shape
Pressure load and erosive burning
Changing shape
Microstructure deformation
Computational Fluid Dynamic (CFD)
Large deformation
Pressure load
Computational Solid Mechanics (CSM)

Burning rate

$$r_b = r_p + r_e + r_s$$

These models are semi-empirical

Basic burning rate

$$r_p = r_{ref}\left(\frac{p}{p_{ref}}\right)^n$$

Erosive burning rate

$$r_e = A\frac{G^{0.8}}{L^{0.2}}e^{-B\rho_p r/G}$$

1957, Lenoir and Robillard

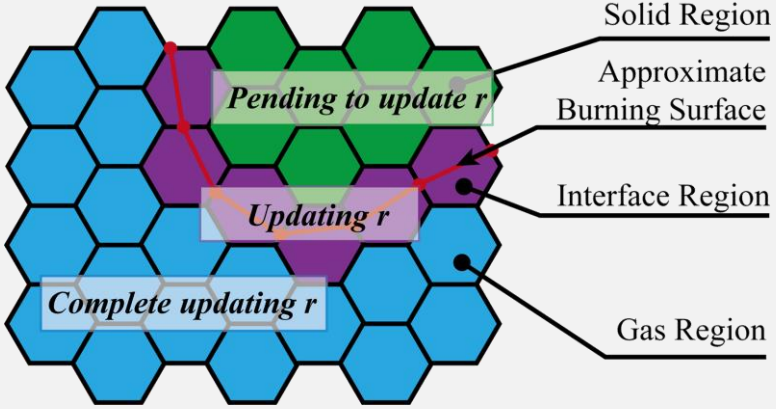
OR

$$r_e = Ap^B\left(u_x - u_{th}\right)^C$$

1980, Razdan and Kuo

Burning rate enhanced by strain

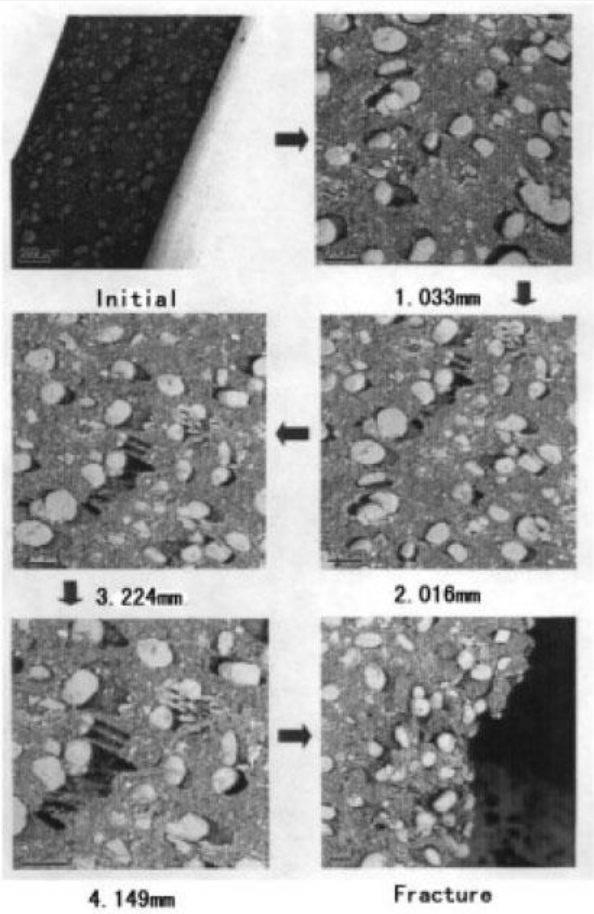
$$r_s = r_0\left(A\varepsilon_3 + B\varepsilon_3^{\,2}\right)$$



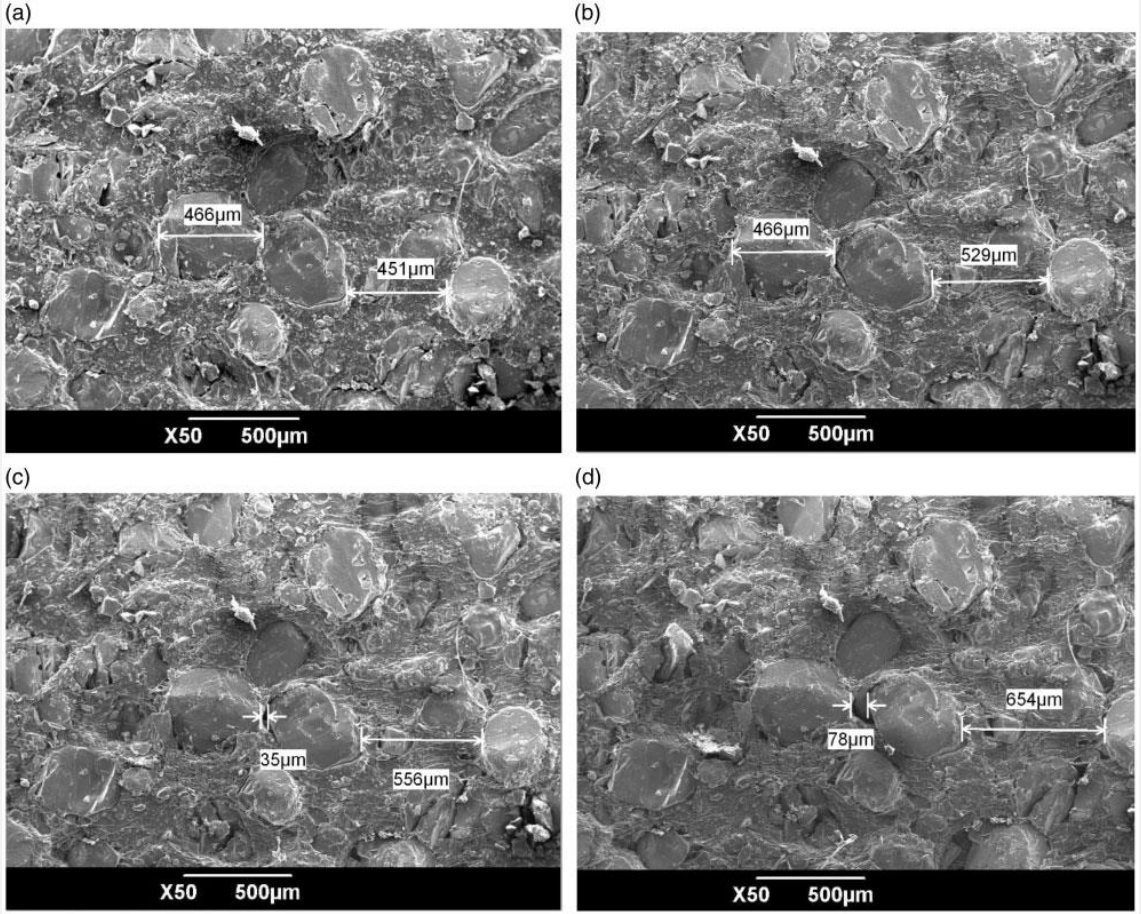
- Burning rate can be directly calculated for every cell at the interface region
- Burning rate at cells that pending to update can be calculated by

Burning Rate Extension Equation

$$\nabla\cdot\left(\left(1-\eta_{rb}\right)h^2\nabla r_b\right) = \eta_{rb}\left(r_b - r_b^{old}\right)$$



The entire process of a typical simple from stretch to destruction.



SEM of propellant under tensile strain: (a) SEM micrographs of 0% deformation; (b) SEM micrographs of 17% deformation; (c) SEM micrographs of 23% deformation; (d) SEM micrographs of 45% deformation.

BSR

$$\nabla\cdot\left(\varepsilon_{\mathrm{r}}\eta_0 r_{\mathrm{b}} h\nabla t_{\mathrm{b}}\right)-\frac{t_{\mathrm{b}}}{\tau}=\eta_0\left(r_{\mathrm{b}}{}^2\nabla t_{\mathrm{b}}{}^{\mathrm{old}}\cdot\nabla t_{\mathrm{b}}{}^{\mathrm{old}}-1-\frac{t_{\mathrm{b}}{}^{\mathrm{old}}}{\tau}\right)$$

$$\nabla\cdot\left(\left(1-\eta_{\mathrm{rb}}\right)h^2\nabla r_{\mathrm{b}}\right)=\eta_{\mathrm{rb}}\left(r_{\mathrm{b}}-r_{\mathrm{b}}^{\mathrm{old}}\right)$$

CFD

$$\begin{cases}\dfrac{\partial\rho}{\partial t}+\nabla\cdot\left(\rho\boldsymbol{v}\right)=S_{\mathrm{m}}\\[2ex]\dfrac{\partial\left(\rho\boldsymbol{v}\right)}{\partial t}+\nabla\cdot\left(\rho\boldsymbol{v}\boldsymbol{v}\right)=-\left(1-\eta_{\mathrm{s}}\right)\nabla p+\nabla\cdot\left(\mu^{\mathrm{e}}\nabla\boldsymbol{v}\right)+\dfrac{1}{3}\nabla\left(\mu^{\mathrm{e}}\nabla\cdot\boldsymbol{v}\right)+S_v\\[2ex]\dfrac{\partial\left(\rho h^*\right)}{\partial t}+\nabla\cdot\left(\rho\boldsymbol{v}h^*\right)-\dfrac{\mathrm{d}p}{\mathrm{d}t}=\nabla\cdot\left(\boldsymbol{\tau}\boldsymbol{v}\right)+\nabla\cdot\left(\lambda^{\mathrm{e}}\nabla T\right)+S_h\end{cases}$$

$$S_{\mathrm{m}}=\frac{r_{\mathrm{b}}\rho_{\mathrm{p}}\left\|\boldsymbol{A}\right\|}{V}$$

$$S_v=-\frac{\left(r_{\mathrm{b}}\rho_{\mathrm{p}}\right)^2\boldsymbol{A}}{V\rho}-\kappa_v\eta_{\mathrm{s}}\boldsymbol{v}$$

$$S_h=S_{\mathrm{m}}\Delta h$$

CSM

$$\nabla\cdot\left(\theta\left(2\mu+\lambda\right)\nabla\boldsymbol{u}_{\mathrm{p}}\right)+\boldsymbol{f}+\nabla\cdot\left(\theta\left(\mu(\nabla\boldsymbol{u}_{\mathrm{p}}{}^{\mathrm{old}})^{\mathrm{T}}+\lambda\mathbf{I}\,\mathrm{tr}(\nabla\boldsymbol{u}_{\mathrm{p}}{}^{\mathrm{old}})-\left(\mu+\lambda\right)\nabla\boldsymbol{u}_{\mathrm{p}}{}^{\mathrm{old}}\right)\right)=0$$

$$\boldsymbol{f}\left(x_{\mathrm{GC},i}\right)=\boldsymbol{f}\left(x_{\mathrm{SC},i}\right)=\frac{p\left(x_{\mathrm{GC},i}\right)\boldsymbol{A}_i}{V_{\mathrm{GC},i}+V_{\mathrm{GS},i}},\quad i\in\mathrm{AI}$$

北京航空航天大學
BEIHANG UNIVERSITY

Content

What is OpenFOAM

- OpenFOAM is the leading free, open source software for CFD owned by the OpenFOAM Foundation/OpenCFD Ltd and distributed exclusively under the General Public Licence (GPL).
- OpenFOAM was first created by Henry Weller in 1989 under the name "FOAM"
- It was released open source as "OpenFOAM" by Henry Weller, Chris Greenshields and Mattijs Janssens in December 2004.

OpenFOAM
The OpenFOAM Foundation

Branch1: openfoam.org

OpenFOAM®

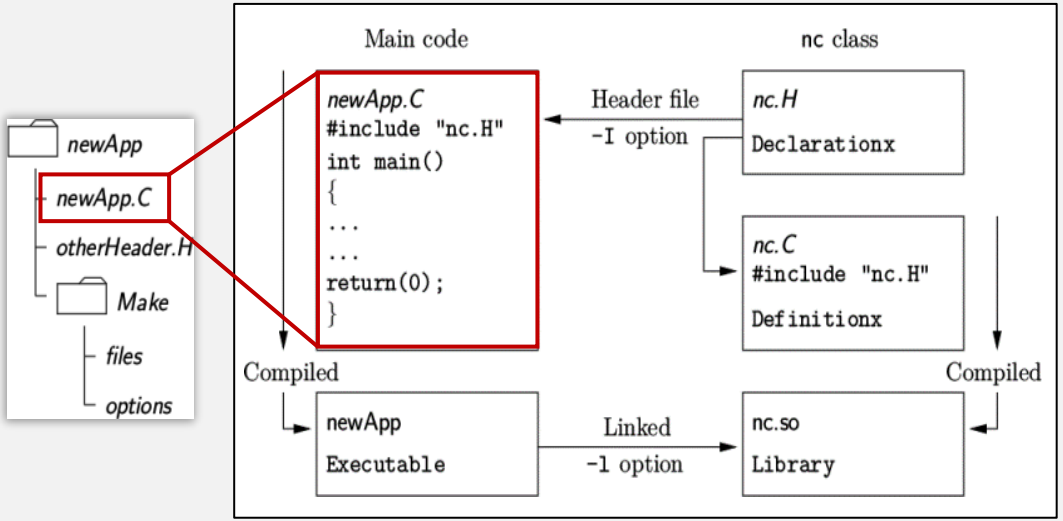
Branch2: www.openfoam.com

Strength of OpenFOAM

In-house code ⟷ Open source code ⟷ Commercial software

Easy to customize | Somewhere between them ... | Fully tested by users

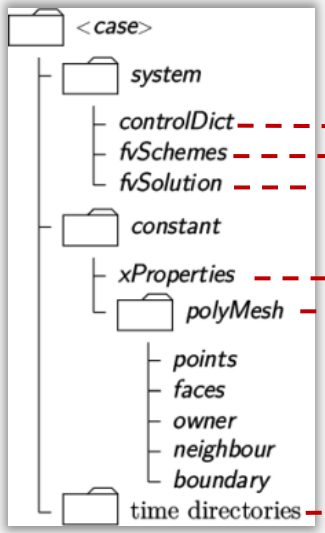
For solver developers



- Write some code in the main function of newApp.C file
- Compile the source code with "wmake" command (an enhanced Linux make command)
- Link the libraries as need

For solver users

- Users can execute the solver under the root directory of the case
- OpenMPI will finish parallel computation



OpenFOAM approximately owns 150 applications and 150 libraries.
Dozens of ready-made solvers are available for reference

Philosophy of OpenFOAM

Field Operation And Manipulation

- Create fields according to the mesh
- Do some computation with fields (Assignment, interpolation, or equation solving)
- Output the fields as wish

Example

```
volVectorField U
(
    IOobject
    (
        "U",
        runTime.timeName(),
        mesh,
        IOobject::MUST_READ,
        IOobject::AUTO_WRITE
    ),
    mesh
);
```

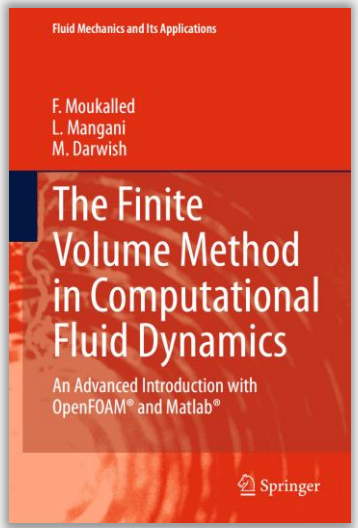
```
fvVectorMatrix UEqn
(
    fvm::ddt(U)
    + fvm::div(phi, U)
    - fvm::laplacian(nu, U)
);

solve(UEqn == -fvc::grad(p));
```

```
runTime.write();

Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
    << "  ClockTime = " << runTime.elapsedClockTime() << " s"
    << nl << endl;
```

➢ Recommend books & resources

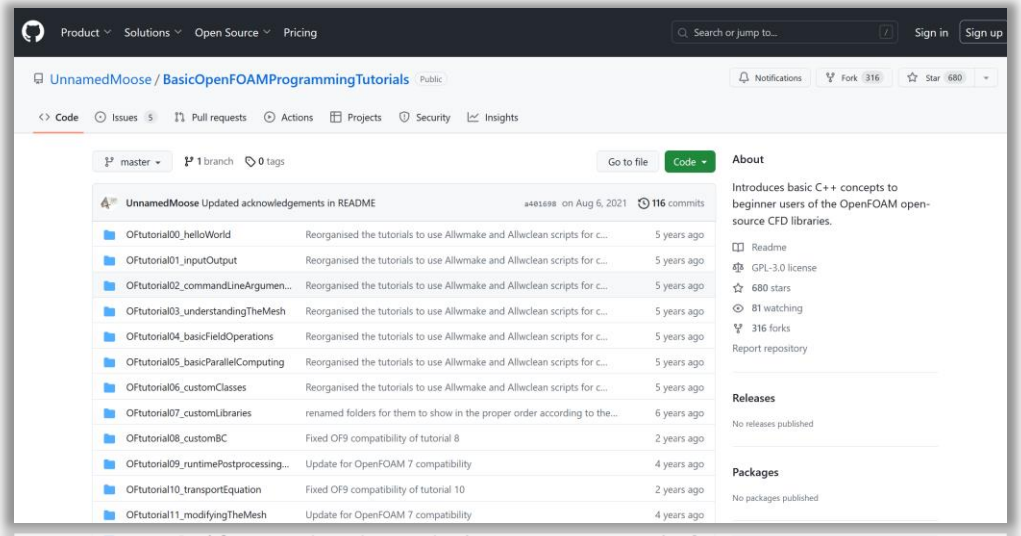


An Advanced Introduction with OpenFOAM and Matlab

by Moukalled 2016

OpenFOAM A little User-Manual

by Holzinger 2023



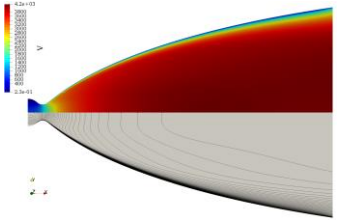
Github: *BasicOpenFOAMProgramingTutorials*

Get hands dirty


SrmFOAM
SOLID ROCKET MOTOR

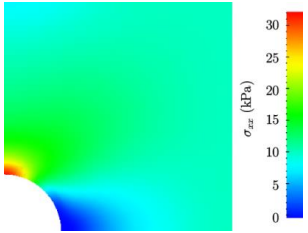
rhoPimpleFoam

Transient solver for turbulent flow of compressible fluids, using a hybrid SIMPLE/PISO algorithm



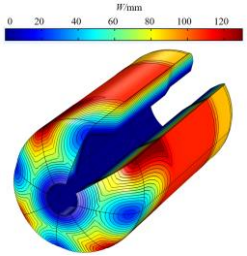
solidDisplacementFoam

Segregated finite-volume solver of linear-elastic, small-strain deformation of a solid body



pefFoam

In-house developed solver for burning surface regression with linear eikonal equation



Code combination

SrmFoam

➢ Continuity equation (pressure correction)

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{v}) = S_{\mathrm{m}}$$

$$S_{\mathrm{m}} = \frac{r_{\mathrm{b}} \rho_{\mathrm{p}} \|\boldsymbol{A}\|}{V}$$

```
volScalarField sourceRho = isBurning * mag(Ac) * rbTemp * rhop / V;
```

In rhoPimpleFoam

```
……
 fvScalarMatrix pDDtEqn
    (
        fvc::ddt(rho) +
psi*correction(fvm::ddt(p))
      + fvc::div(phiHbyA) + fvm::div(phid, p)
     ==
        fvOptions(psi, p, rho.name())
);
……
```

Modified

In srmFoam

```
……
 fvScalarMatrix pDDtEqn
    (
        fvc::ddt(rho) +
psi*correction(fvm::ddt(p))
      + fvc::div(phiHbyA) + fvm::div(phid, p)
     ==
        fvOptions(psi, p, rho.name())
        +sourceRho
    );
……
```

➢ Momentum equation

$$\frac{\partial(\rho \boldsymbol{v})}{\partial t}+\nabla\cdot(\rho \boldsymbol{v}\boldsymbol{v})=-(1-\eta_s)\nabla p+\nabla\cdot\left(\mu^e\nabla \boldsymbol{v}\right)+\frac{1}{3}\nabla\left(\mu^e\nabla\cdot \boldsymbol{v}\right)+S_v$$

$$S_v=-\frac{\left(r_b\rho_p\right)^2 \boldsymbol{A}}{V\rho}-\kappa_v\eta_s\boldsymbol{v}$$

```
volVectorField sourceU = -isBurning * rhop *
rhop * rbTemp * rbTemp * Ac / (V * rho);
```

In rhoPimpleFoam

```
……
tmp<fvVectorMatrix> tUEqn
(
fvm::ddt(rho, U) + fvm::div(phi, U)
+ MRF.DDt(rho, U)
+ turbulence->divDevTau(U)
==
fvOptions(rho, U)
);
……
solve(UEqn == - fvc::grad(p));
……
```

Modified

In srmFoam

```
……
tmp<fvVectorMatrix> tUEqn
(
fvm::ddt(rho, U) + fvm::div(phi, U)
+ MRF.DDt(rho, U)
+ turbulence->divDevTau(U)
==
fvOptions(rho, U)
+ sourceU
- fvm::Sp( (KU + KUU * mag(U)) * keyS ,U)
);
……
solve(UEqn == - (1 - keyS) *fvc::grad(p));
……
```

Total enthalpy equation

$$\frac{\partial\left(\rho h^{*}\right)}{\partial t}+\nabla\cdot\left(\rho \boldsymbol{v} h^{*}\right)-\frac{\mathrm{d}p}{\mathrm{d}t}=\nabla\cdot\left(\boldsymbol{\tau}\boldsymbol{v}\right)+\nabla\cdot\left(\lambda^{\mathrm{e}}\nabla T\right)+S_h$$

$$S_h = S_{\mathrm{m}}\Delta h$$

```
volScalarField sourceE = sourceRho * deltaH;
```

```
……
fvScalarMatrix EEqn
    (
        fvm::ddt(rho, he) +  fvm::div(phi, he)
      + fvc::ddt(rho, K) + fvc::div(phi, K)
      -dpdt
      + thermophysicalTransport->divq(he)
     ==
        fvOptions(rho, he)
);
……
```

Modified

```
……
fvScalarMatrix EEqn
    (
        fvm::ddt(rho, he) +  fvm::div(phi, he)
      + fvc::ddt(rho, K) + fvc::div(phi, K)
      -dpdt
      + thermophysicalTransport->divq(he)
     ==
        fvOptions(rho, he)
      + sourceE
);
……
```

Deformation equation

$$\nabla\cdot\left(\theta\left(2\mu+\lambda\right)\nabla\boldsymbol{u}_{\mathrm{p}}\right)+\boldsymbol{f}+\nabla\cdot\left(\theta\left(\mu(\nabla\boldsymbol{u}_{\mathrm{p}}^{\mathrm{old}})^{\mathrm{T}}+\lambda\mathbf{I}\,\mathrm{tr}(\nabla\boldsymbol{u}_{\mathrm{p}}^{\mathrm{old}})-\left(\mu+\lambda\right)\nabla\boldsymbol{u}_{\mathrm{p}}^{\mathrm{old}}\right)\right)=0$$

```
……
do
{
    {
        gradD = factor * fvc::grad(D);
        sigmaD = (mu * twoSymm(gradD) + (lambda * I) * tr(gradD));
        divSigmaExp = fvc::div(sigmaD - (2 * mu + lambda) * gradD, "div(sigmaD)");
    }
    {
        D.storePrevIter();
        fvVectorMatrix DEqn(fvm::laplacian(factor * (2 * mu + lambda), D, "laplacian(DD,D)") + divSigmaExp + sourceF);
        fvOptions.constrain(DEqn);
        initialResidual = DEqn.solve().max().initialResidual();
        D.relax(0.2);
    }
} while (initialResidual > convergenceTolerance && ++iCorr < nCorr);

……
```

Linearized eikonal equation

$$\nabla\cdot\left(\varepsilon_r\eta_0 r_b h\nabla t_b\right)-\frac{t_b}{\tau}=\eta_0\left(r_b{}^2\nabla t_b{}^{old}\cdot\nabla t_b{}^{old}-1-\frac{t_b{}^{old}}{\tau}\right)$$

```
……
volScalarField tao = 0.2 * h / rb;
volVectorField gradTb = fvc::grad(tb);
volScalarField Su = rb * rb * (gradTb & gradTb) - 1 - tb / tao;
volScalarField Sp = 1.0 / tao;

while (simple.correctNonOrthogonal())
{
    fvScalarMatrix tbEqn(
        fvm::laplacian(keyS0 * h * rb, tb) == keyS0 * Su +
fvm::Sp(Sp, tb));
    tbEqn.solve();
}

……
```

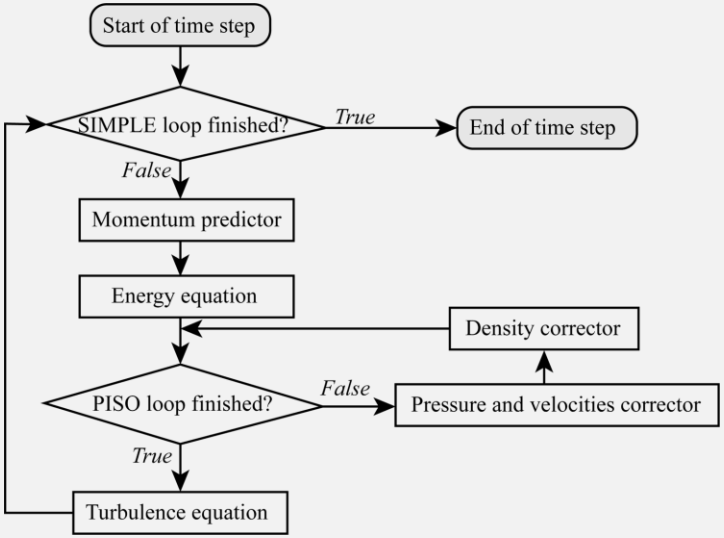
Burning rate extension equation

$$\nabla\cdot\left(\left(1-\eta_{rb}\right)h^2\nabla r_b\right)=\eta_{rb}\left(r_b-r_b^{old}\right)$$

```
……
while (simple.correctNonOrthogonal())
{
  fvScalarMatrix rbEqn(
        fvm::laplacian((1.0 - keyR) * h * h, rb)
== fvm::Sp(keyR, rb) - keyR * rb);
    rbEqn.solve();
}
……
```

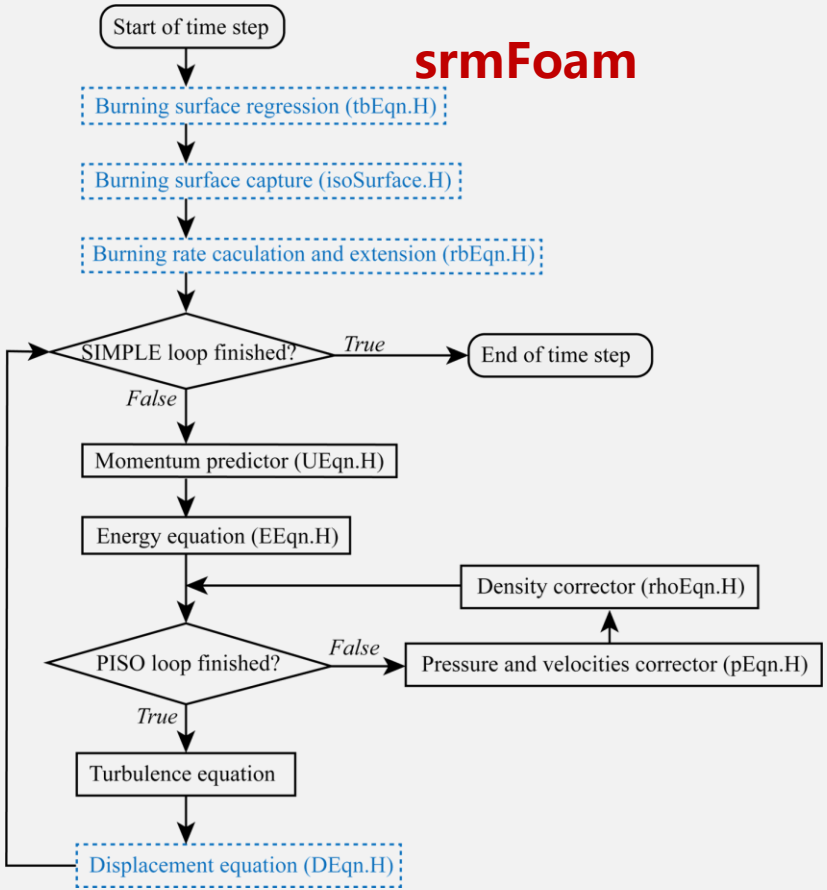
➢ Flow chart

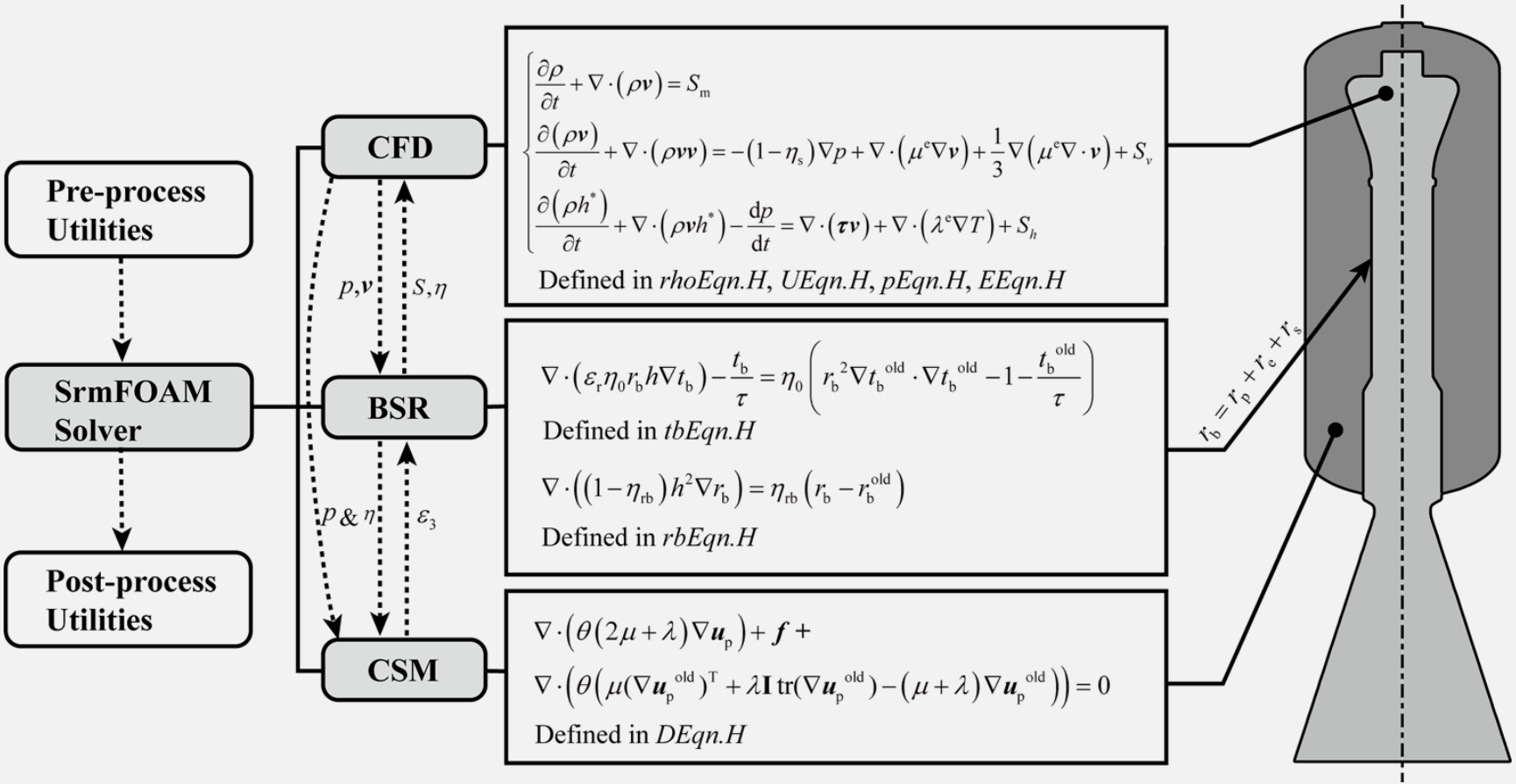
rhoPimpleFoam



Modified

srmFoam




Pre-process Utilities
SrmFOAM Solver
Post-process Utilities
CFD
BSR
CSM
p,v
S,η
p & η
ε3
$\frac{\partial \rho}{\partial t}+\nabla\cdot(\rho \boldsymbol{v})=S_{\mathrm{m}}$
$\frac{\partial(\rho \boldsymbol{v})}{\partial t}+\nabla\cdot(\rho \boldsymbol{v}\boldsymbol{v})=-(1-\eta_{\mathrm{s}})\nabla p+\nabla\cdot\left(\mu^{\mathrm{e}}\nabla \boldsymbol{v}\right)+\frac{1}{3}\nabla\left(\mu^{\mathrm{e}}\nabla\cdot \boldsymbol{v}\right)+S_{v}$
$\frac{\partial(\rho h^{*})}{\partial t}+\nabla\cdot\left(\rho \boldsymbol{v} h^{*}\right)-\frac{\mathrm{d}p}{\mathrm{d}t}=\nabla\cdot(\boldsymbol{\tau}\boldsymbol{v})+\nabla\cdot\left(\lambda^{\mathrm{e}}\nabla T\right)+S_{h}$
Defined in rhoEqn.H, UEqn.H, pEqn.H, EEqn.H
$\nabla\cdot\left(\varepsilon_{\mathrm{r}}\eta_{0}r_{\mathrm{b}}h\nabla t_{\mathrm{b}}\right)-\frac{t_{\mathrm{b}}}{\tau}=\eta_{0}\left(r_{\mathrm{b}}{}^{2}\nabla t_{\mathrm{b}}{}^{\mathrm{old}}\cdot\nabla t_{\mathrm{b}}{}^{\mathrm{old}}-1-\frac{t_{\mathrm{b}}{}^{\mathrm{old}}}{\tau}\right)$
Defined in tbEqn.H
$\nabla\cdot\left(\left(1-\eta_{\mathrm{rb}}\right)h^{2}\nabla r_{\mathrm{b}}\right)=\eta_{\mathrm{rb}}\left(r_{\mathrm{b}}-r_{\mathrm{b}}^{\mathrm{old}}\right)$
Defined in rbEqn.H
$\nabla\cdot\left(\theta\left(2\mu+\lambda\right)\nabla \boldsymbol{u}_{\mathrm{p}}\right)+\boldsymbol{f}+$
$\nabla\cdot\left(\theta\left(\mu(\nabla \boldsymbol{u}_{\mathrm{p}}{}^{\mathrm{old}})^{\mathrm{T}}+\lambda\mathbf{I}\,\mathrm{tr}(\nabla \boldsymbol{u}_{\mathrm{p}}{}^{\mathrm{old}})-(\mu+\lambda)\nabla \boldsymbol{u}_{\mathrm{p}}{}^{\mathrm{old}}\right)\right)=0$
Defined in DEqn.H
$r_{\mathrm{b}}=r_{\mathrm{p}}+r_{\mathrm{e}}+r_{\mathrm{s}}$

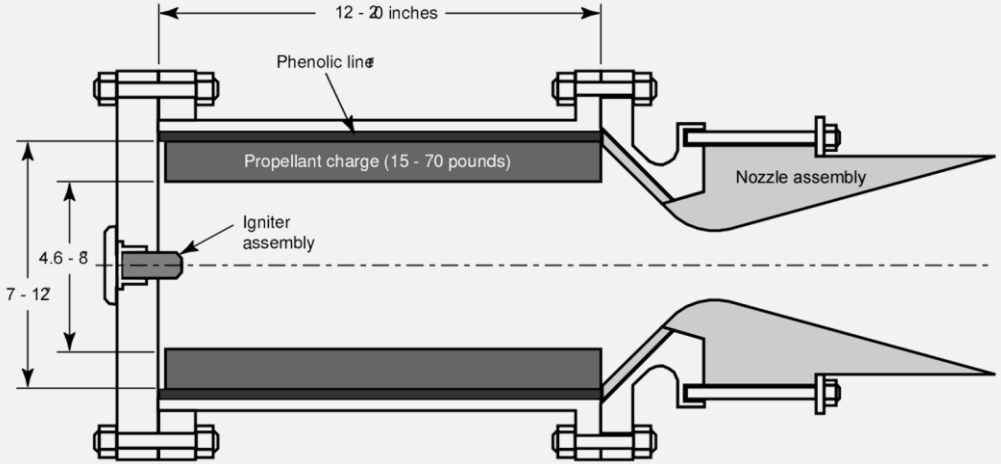
北京航空航天大學
BEIHANG UNIVERSITY

Content

BATES Motor

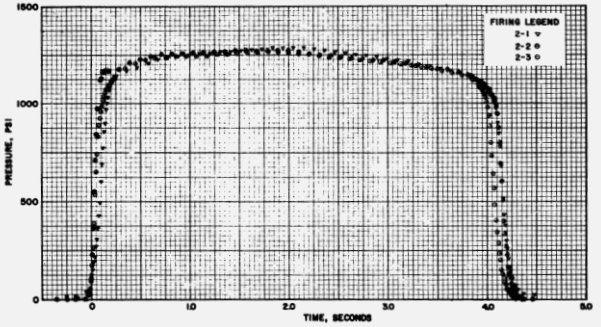
➢ BATES motor

- The Ballistic Test and Evaluation System (BATES) is a series of solid rocket motors developed by Air Force Rocket Propulsion Laboratory (AFRPL)
- It provides a lot of valuable experimental data to investigate the solid rocket motors

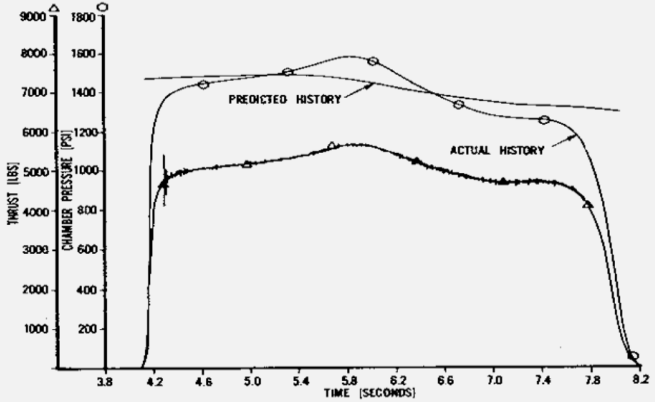


Propellant components: 69% AP+ 21% Al + 10% HTPB

➢ Normal firing test



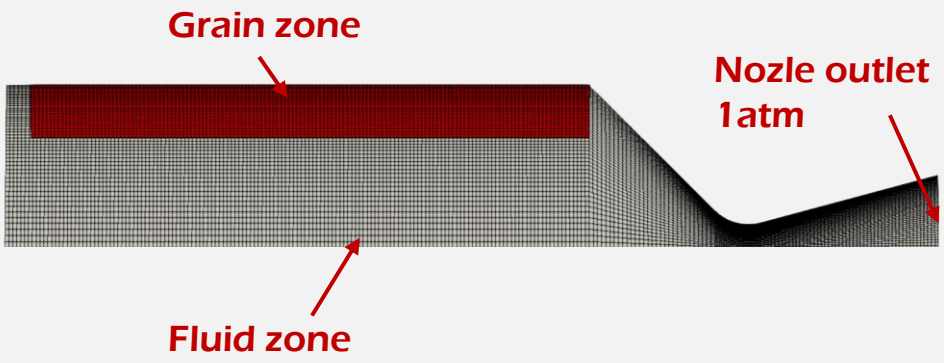
➢ Anomaly pressure peak



BATES Motor

Geometry parameters

Items	Value	Unit
Grain outer radius	5.875	inch
Grain length	20	inch
Nozzle throat radius	0.85	inch
Nozzle convergence angle (half)	45	deg
Nozzle expansion angle (half)	15	deg
Nozzle expansion area ratio	9.5	



Physical parameters

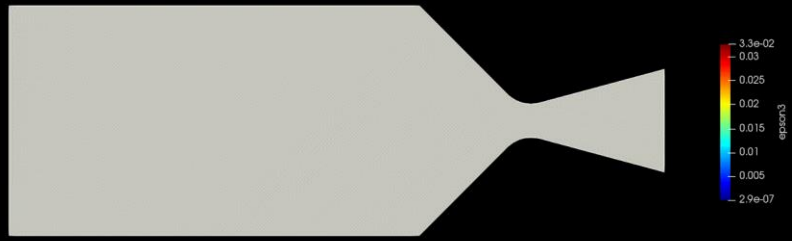
Items	Value	Unit
Propellant density	1667	kg/m^3
Adiabatic combustion temperature	3746	K
Gas molecular weight	30.646	g/mol
Gas specific heat capacity	1803.85	J/(kg-K)
Dynamic viscosity	9.8868E-5	Pa-s
Plante number	0.449	
Reference pressure	8.1358	MPa
Reference burning rate	11.9063	mm/s
Pressure index	0.3	
Elastic modulus	3.5	MPa
Poisson's ratio	0.498	

※ Burned gas property are calculated by chemical equilibrium method

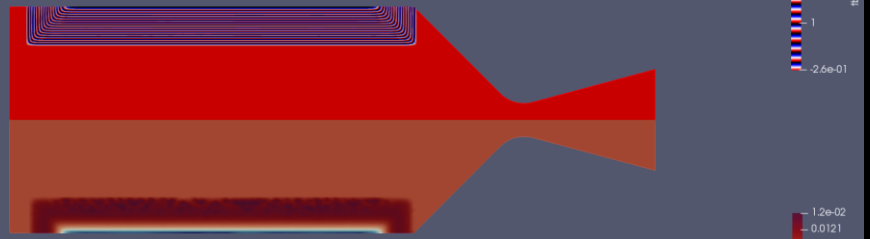
➢ Simulation results

- The strain/stress field is obtained as well as flow field. Maximum strain reaches 0.33.
- The burned time field shows the process of the entire burning. The burning stops at 3.9 sec. The burning rate approximately remains constant.
- The simulation provide a full physical picture during the burning.

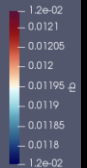
Strain field (0~4.5s)



Burned time field

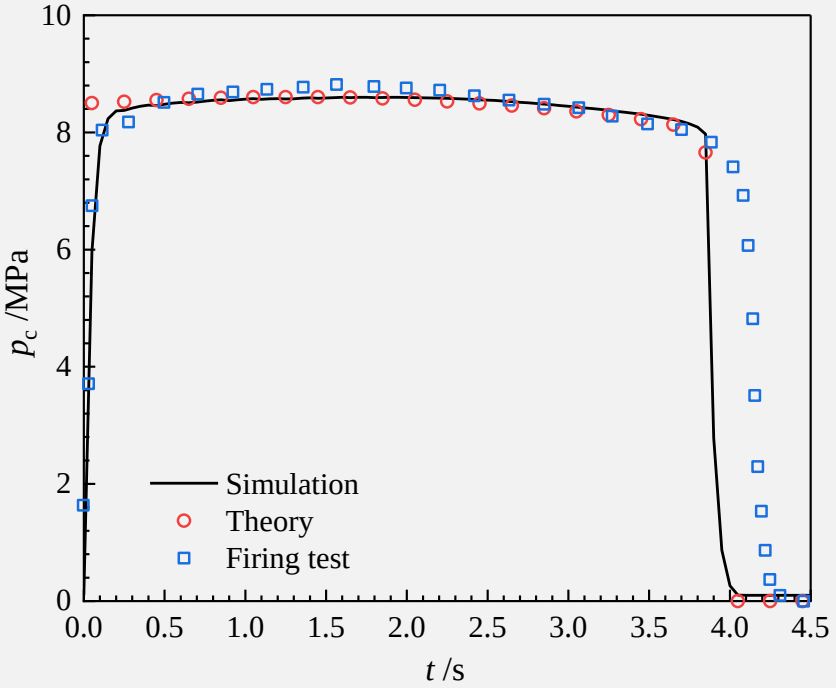


Burning rate field



Mach number field (0~4.5s)

➢ Internal ballistic performance



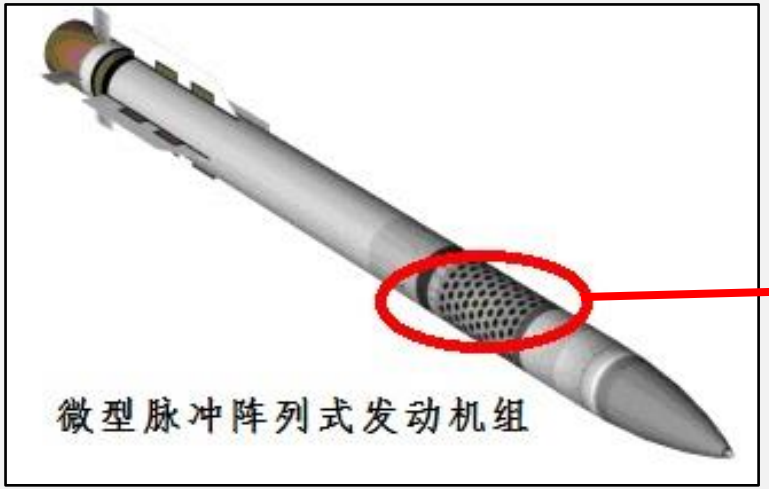
- The simulation results is very close to the theoretical solution.
- The simulation match well with the firing test at the beginning, but estimate the end time of burning too early.
- The reason may come from the inaccurate calculation of the burning rate during the pressure drop phase.
- The reason may also come from the inaccuracy of the given propellant density and reference burning rate.

(burning rate may be faster than the real motor and propellant density is lower than the real one)

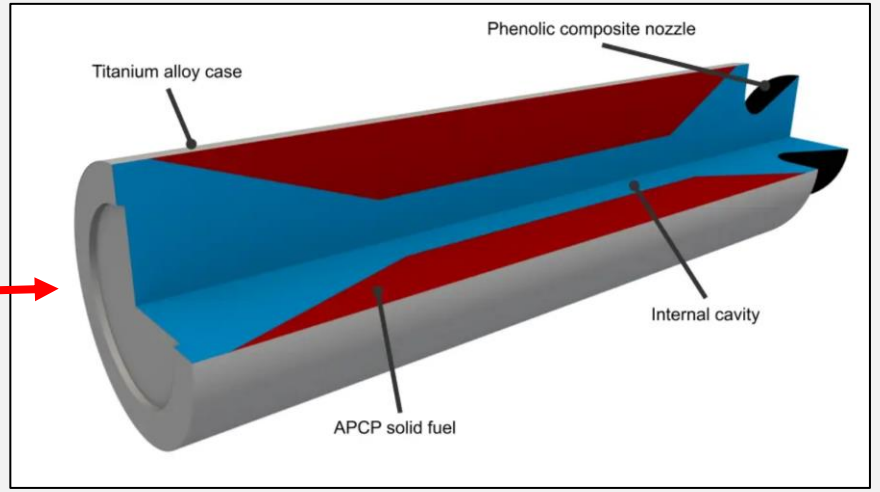
ACM Case

➢ Attitude control motor

- The Attitude Control Motor (ACM) is a small solid rocket motor consisting of a titanium alloy case, phenolic composite nozzle, and APCP propellant (ammonium perchlorate composite).
- Arrays of these motors are mounted on larger rockets and used to control the orientation or "attitude" of a missile in the final moments before impact.



Pulsed solid attitude control system (containing hundreds of ACMs)



One attitude control motor

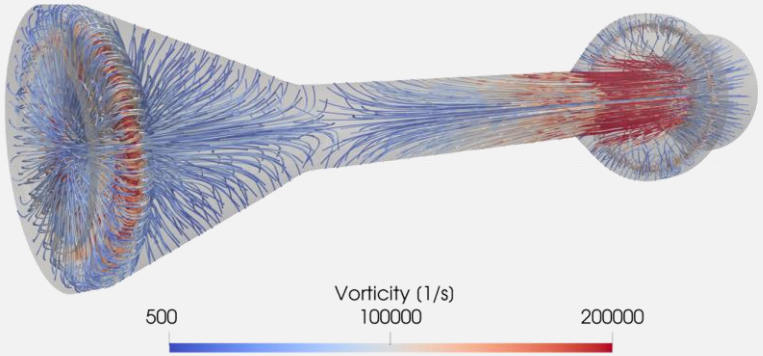
ACM Case

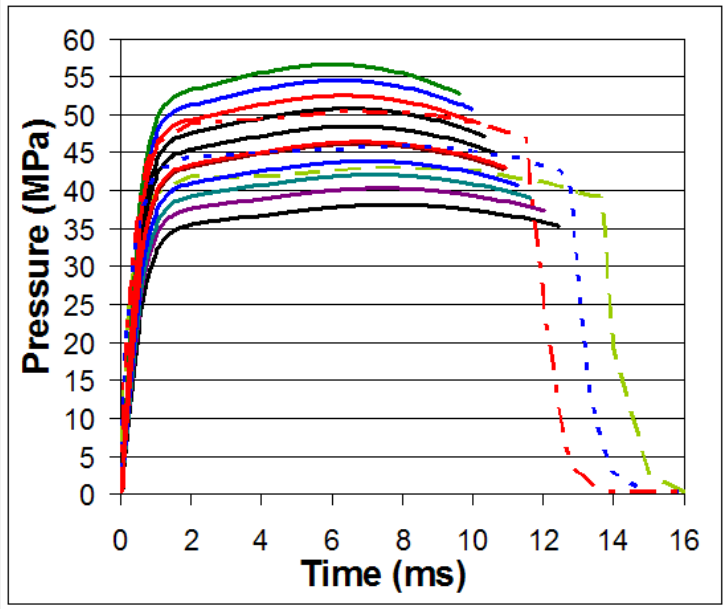
➢ Challenges

- ACM is highly transient. Combustion lasts only 14 ms, and the combustion chamber pressure can reach 40 MPa. The working process is more like a tiny explosion.
- The internal ballistics of ACM is highly sensitive. A small change in the input parameters will result in a change of combustion chamber pressure of more than ten MPa.

➢ Previous work

- ACM was previously studied by Illinois University with their software Rocstar Multiphysics



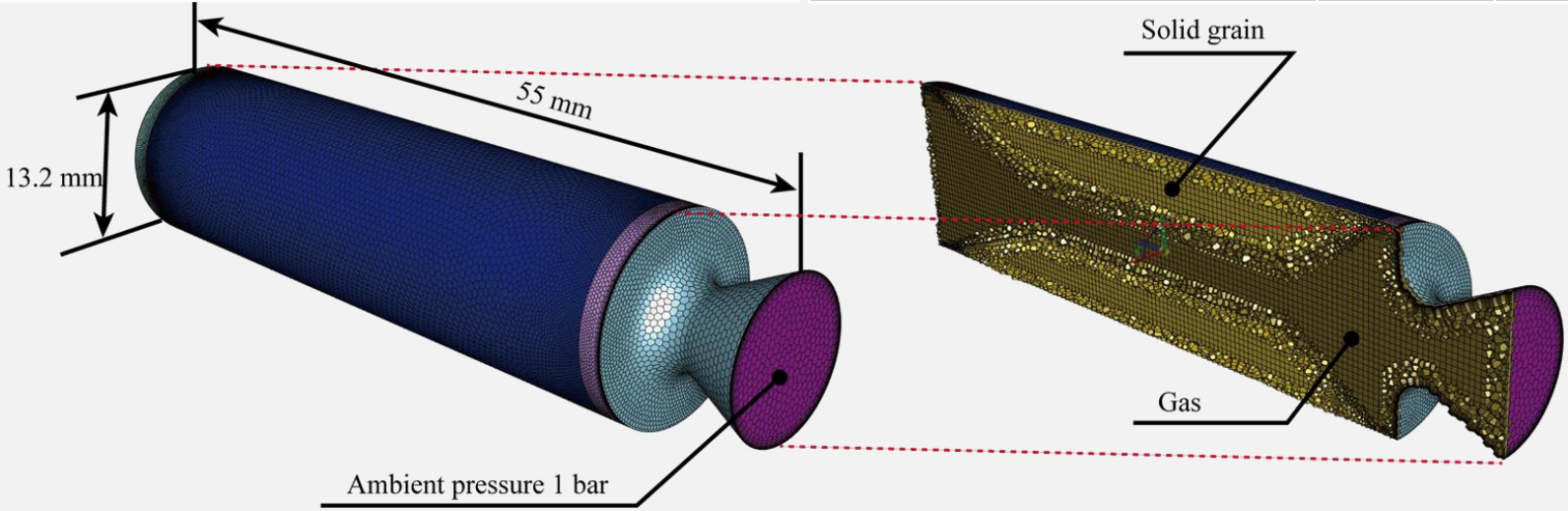


ACM Case

Physical parameters

Items	Value	Unit
Propellant density	1650	kg/m³
Adiabatic combustion temperature	2989.6	K
Gas molecular weight	22.1165	g/mol
Gas specific heat capacity	1952.1	J/(kg-K)

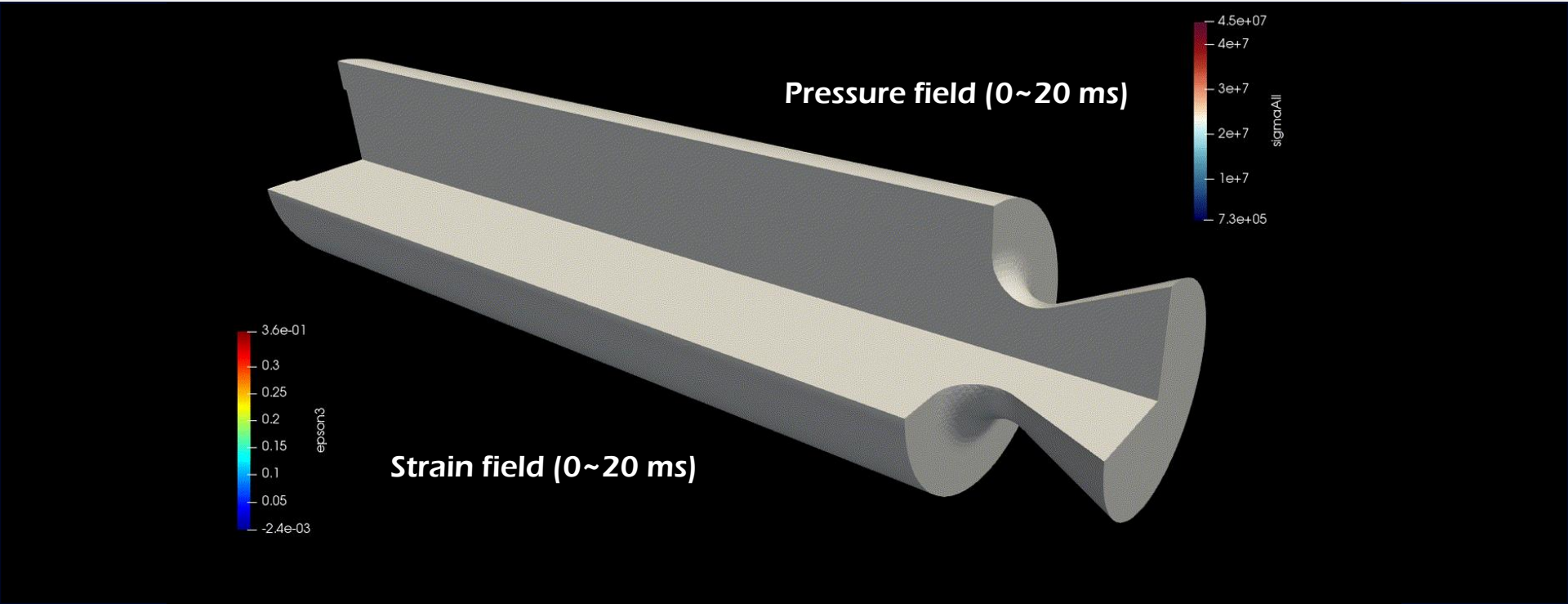
Items	Value	Unit
Reference pressure	0.101325	MPa
Reference burning rate	0.00781	mm/s
Pressure index	0.62	
Elastic modulus	3.5	MPa
Poisson's ratio	0.498	



ACM Case

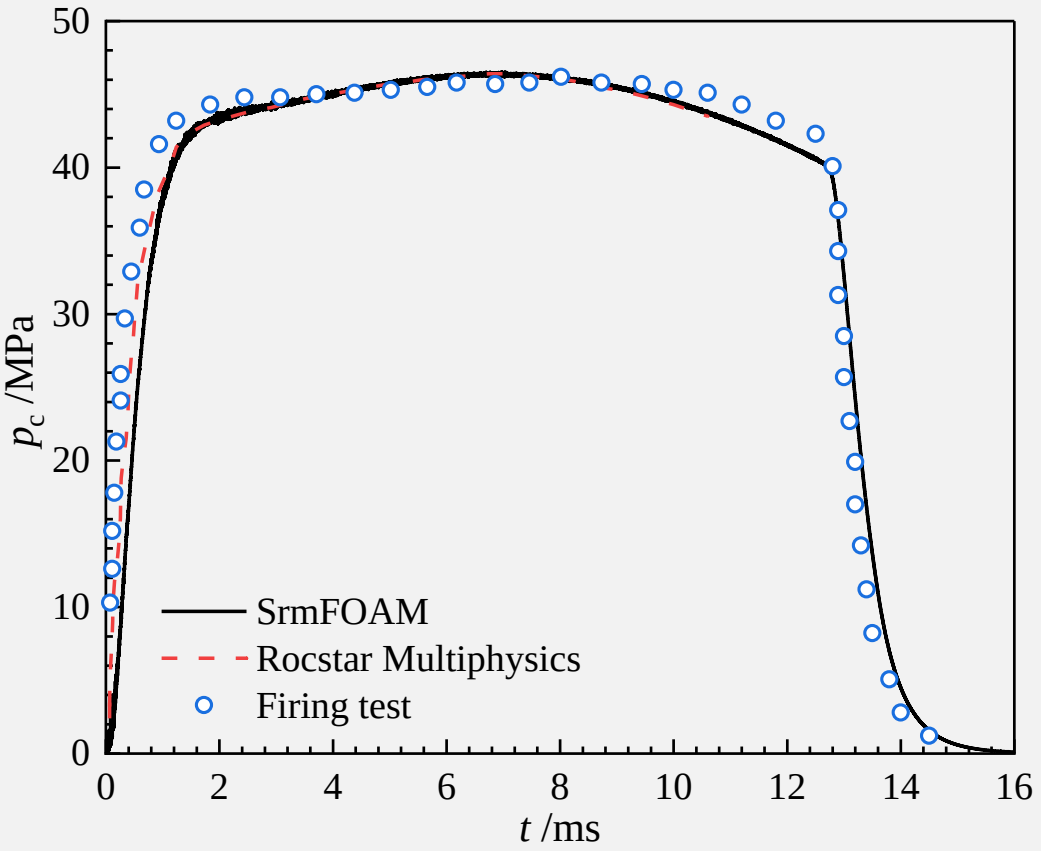
Simulation results

- Max strain happens at the beginning, reaching 0.36.
- There is a pressure drop a the beginning, causing an non-uniformed burning rate.
- The simulation provide a full physical picture during the burning.



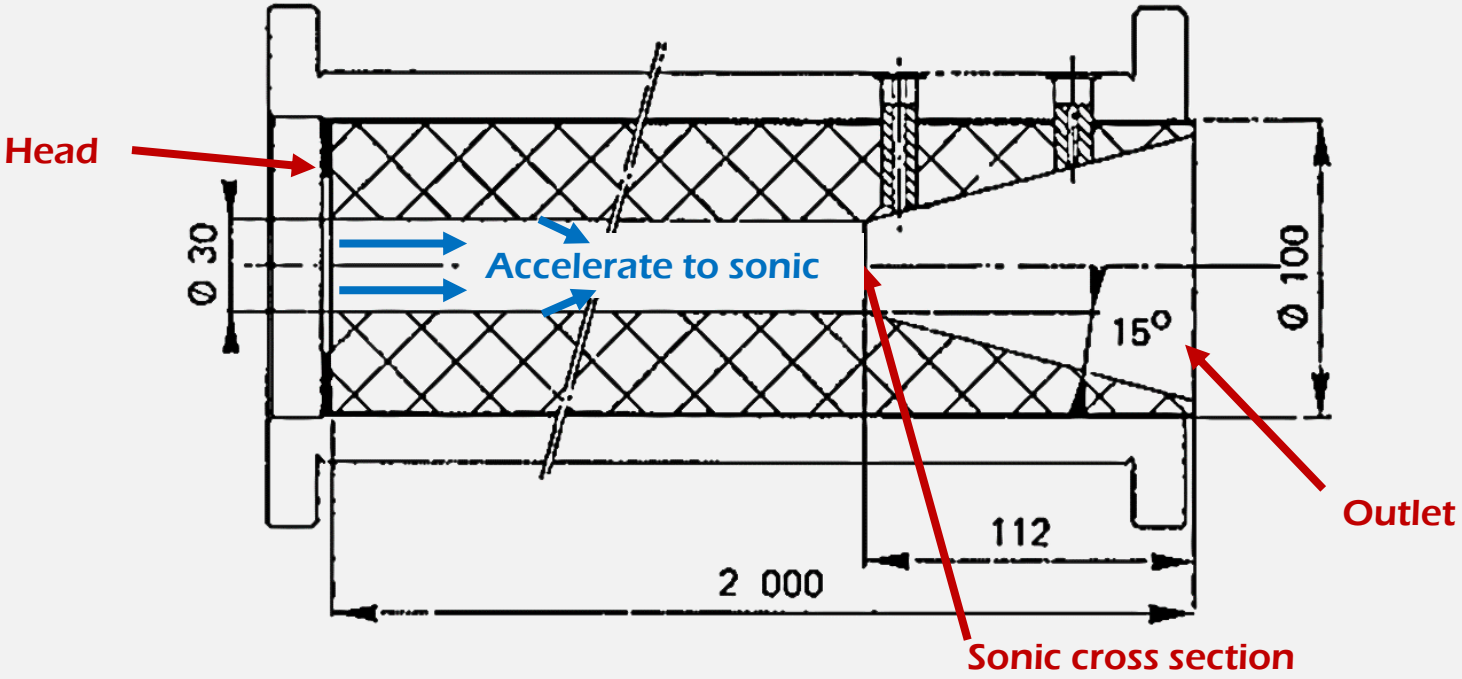
Internal ballistic performance

- The pressure reaches almost 45 MPa and lasts for 14 milliseconds. The pressure rises by approximately 40 MPa within 2 milliseconds.
- SrmFOAM results is similar to the Rocstar Multihysics and firing test.
- The error may come from the lack of ignition model as we did not simulate the ignitor.
- The reason may also come from the inaccurate calculation of the burning rate at large pressure jump.



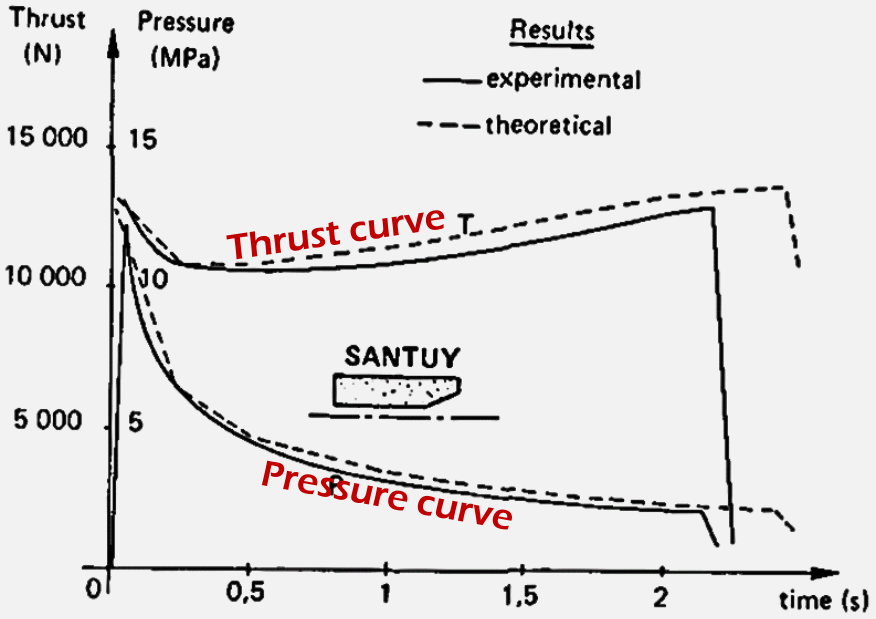
Nozzleless motor

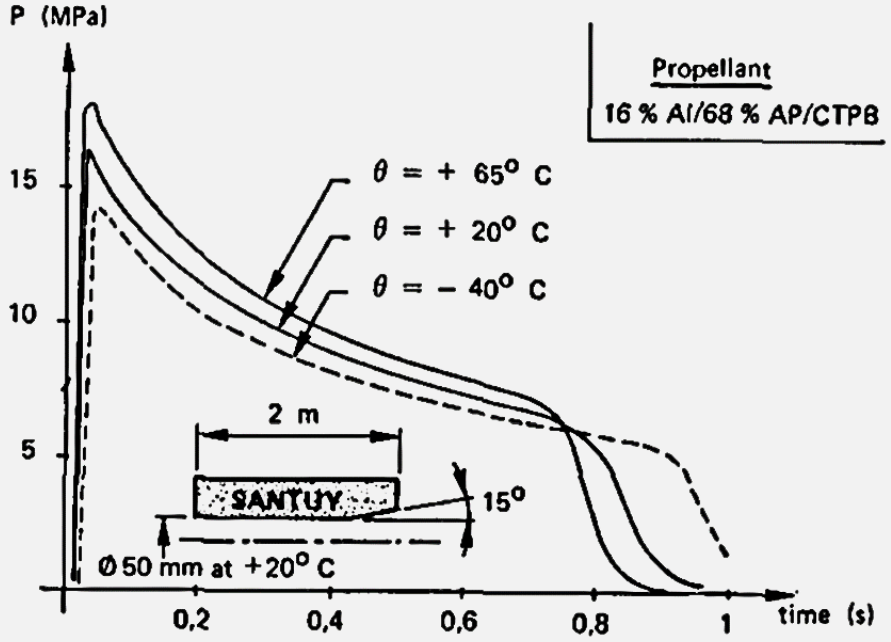
- Nozleless motors do not have nozzle
- They have a good reliability and easy to manufacture
- However, the performance prediction is challenge



➢ Experimental results

- During combustion, the sonic cross section area increases, leading to a decrease in chamber pressure.
- There is no significant change in thrust throughout the process.

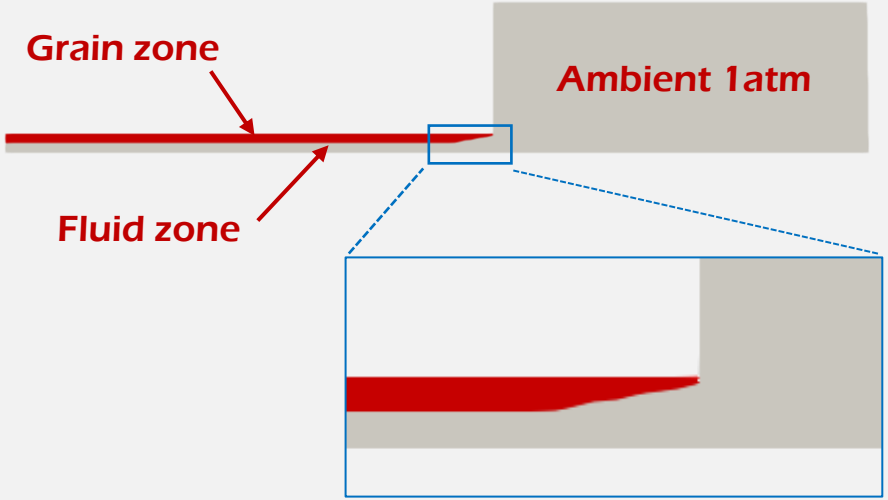




Nozzleless Motor

Geometry parameters

Items	Value	Unit
Grain outer radius	100	mm
Grain length	1300	mm
"Nozzle" expansion angle (half)	15	deg
"Nozzle" expansion length	112	mm



Physical parameters

Items	Value	Unit
Propellant density	1650	kg/m³
Adiabatic combustion temperature	3178	K
Gas molecular weight	26.02	g/mol
Gas specific heat capacity	1895.6	J/(kg-K)
Dynamic viscosity	8.598E-5	Pa-s
Plante number	0.437	
Reference pressure	7.0	MPa
Reference burning rate	25	mm/s
Pressure index	0.23	

※ **Burned gas property are calculated by chemical equilibrium method**

Erosive burning model

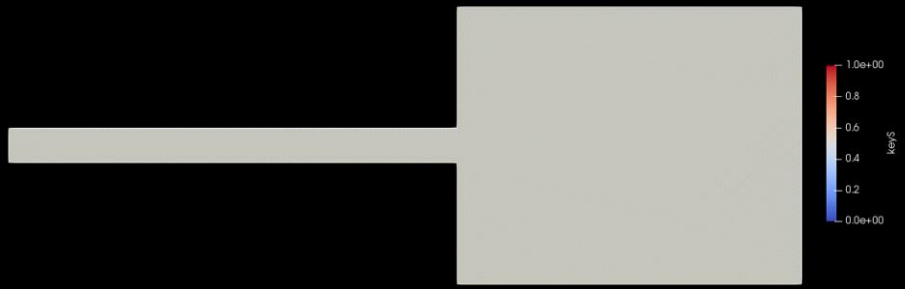
1957, Lenoir and Robillard

$$r_e = A\frac{G^{0.8}}{L^{0.2}}e^{-B\rho_p r/G}$$

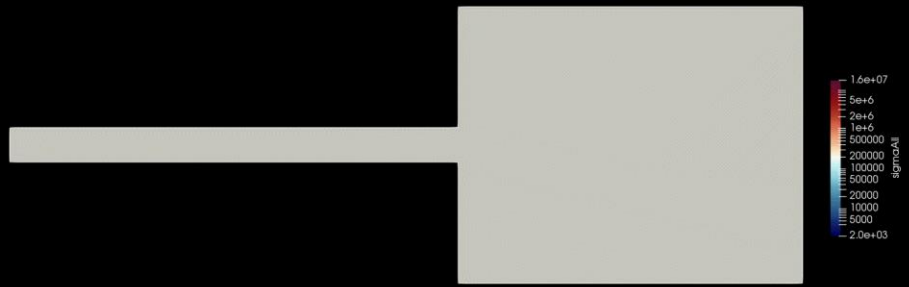
Simulation results

- The erosive burning is obvious during the burning.
- There is a great pressure drop along the port.
- The sonic line is at the beginning of the expansion section.
- Due to the pressure gradient, there are shock waves at the outlet.

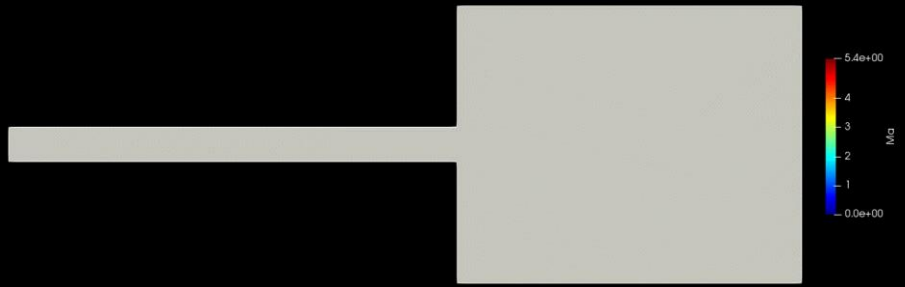
Solid region (0~1.2s)



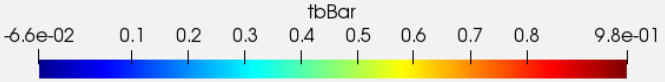
Pressure field (0~1.2s)

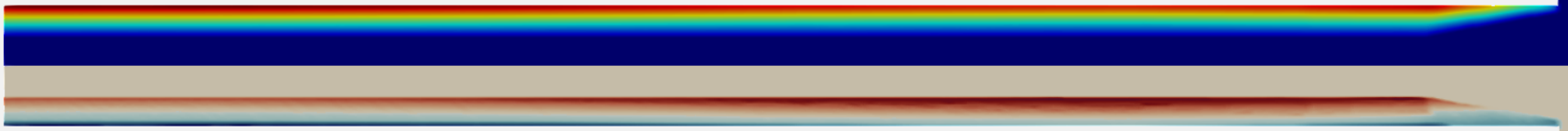


Mach number field (0~1.2s)



Burned time field



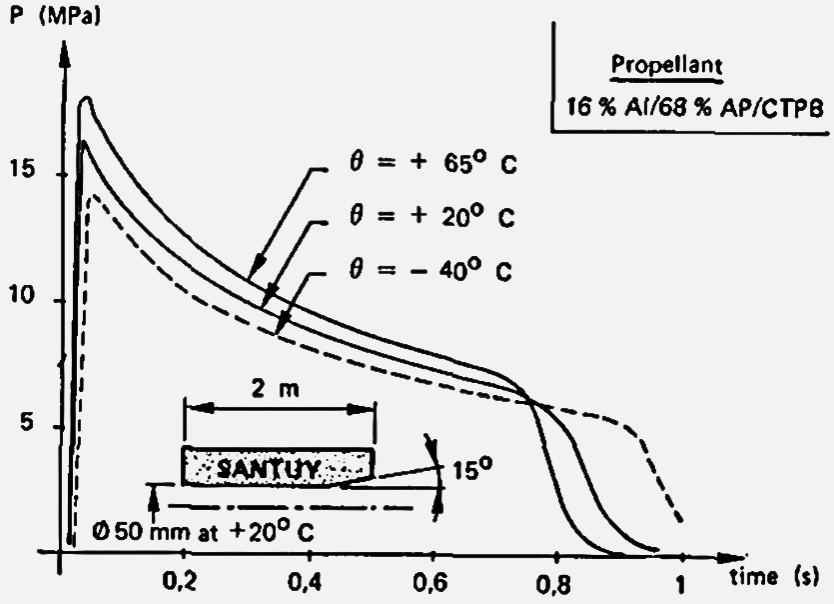


Burning rate field

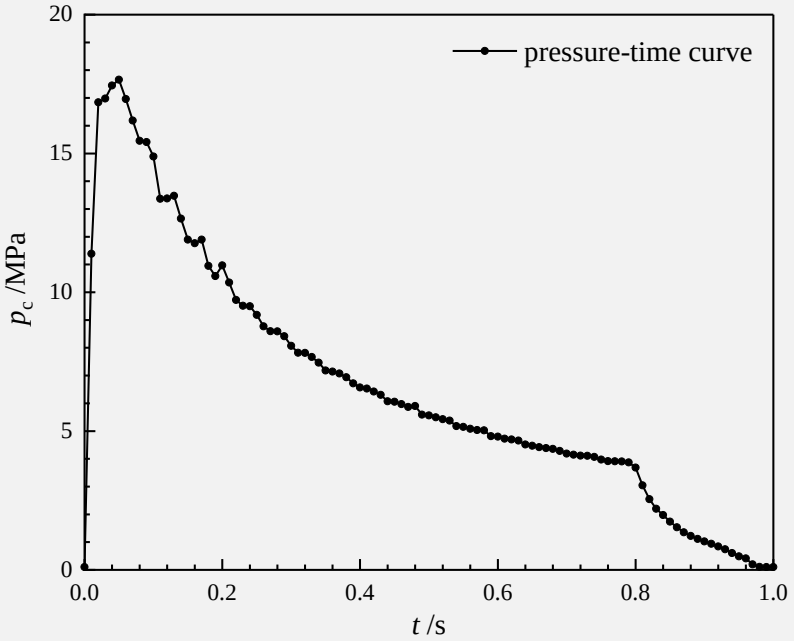
- The burning rate is significantly non-uniform
- In sections with high mass flow density, the burning rate is relatively higher.

Internal ballistic performance

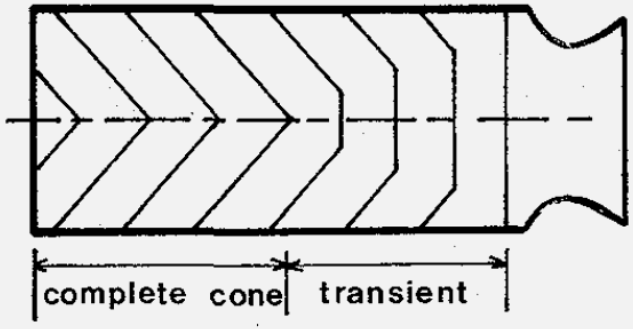
Firing test results

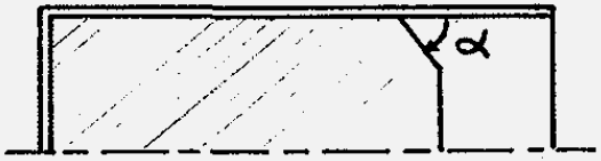


Simulation results at 20 ℃



End-burning Motor





➢ End-burning motor

- Theoretically, end-burning motors have constant burning surface area as well as chamber pressure.
- However, firing test shows that there are "burning rate acceleration" near the wall.

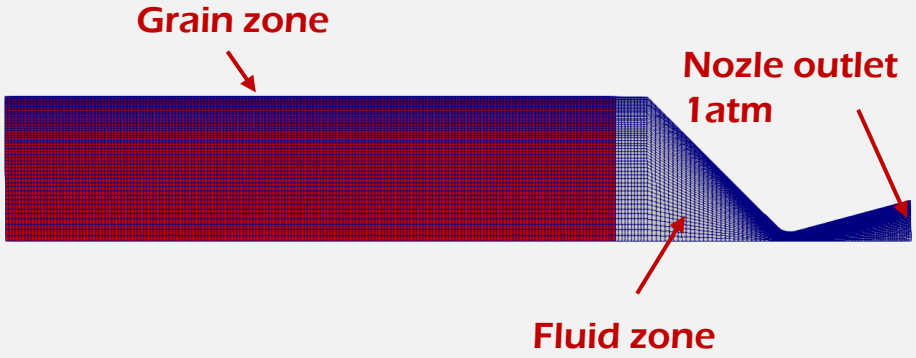
➢ Reason of burning rate acceleration

- Chemically, it is caused by the migration of species from the propellant into the inhibitor
- Mechanically, it is caused by the tiny cracks due to the strain under high pressure

End-burning Motor

Geometry parameters

Items	Value	Unit
Grain outer radius	45	mm
Grain length	190	mm
Nozzle throat radius	0.85	mm
Nozzle convergence angle (half)	45	deg
Nozzle expansion angle (half)	15	deg
Nozzle expansion area ratio	20	

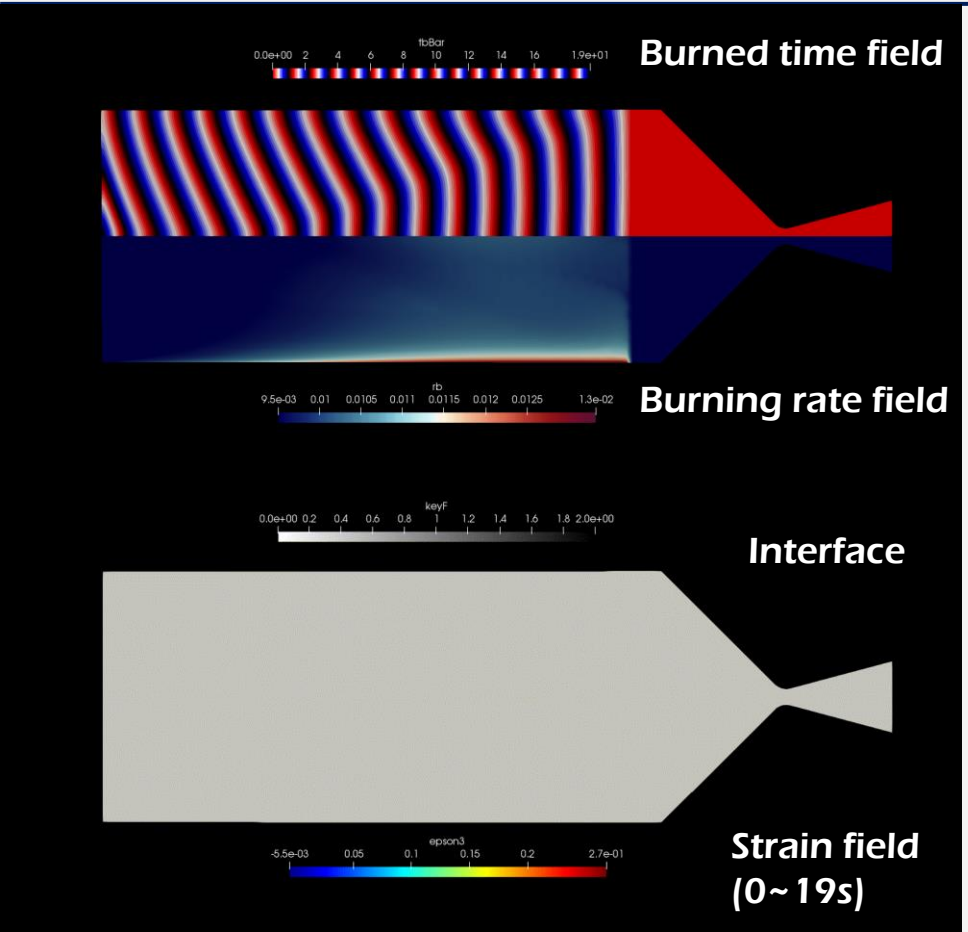


Physical parameters

Items	Value	Unit
Propellant density	1667	kg/m^3
Adiabatic combustion temperature	3746	K
Gas molecular weight	30.646	g/mol
Gas specific heat capacity	1803.85	J/(kg-K)
Dynamic viscosity	9.8868E-5	Pa-s
Plante number	0.449	
Reference pressure	8.1358	MPa
Reference burning rate	11.9063	mm/s
Pressure index	0.3	
Elastic modulus	3.5	MPa
Poisson's ratio	0.498	

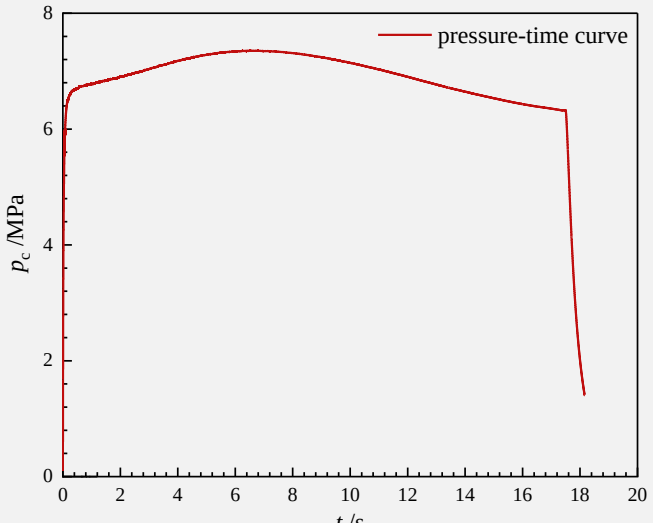
$$r_s = r_0\left(1.34\varepsilon_3 + 0.136{\varepsilon_3}^2\right)$$

End-burning Motor

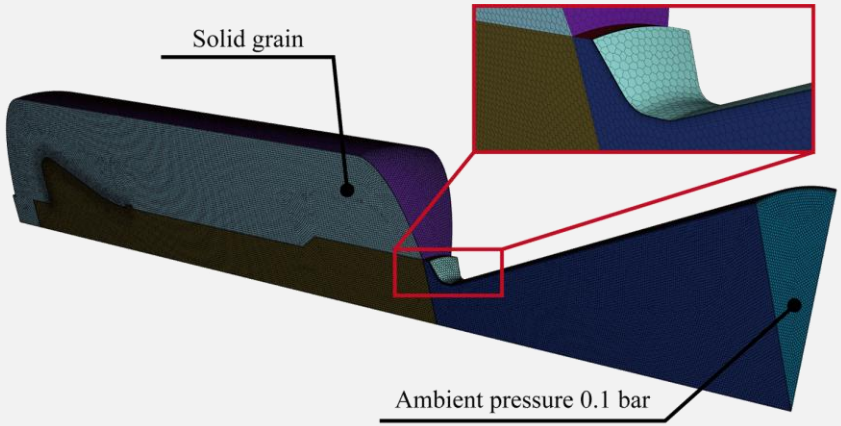


➢ Simulation results

- There is stress concentration at the interface between the propellant and the wall, which leads to an increase in burning rate.
- As time goes on, the burning surface changes from a plane to a cone, and the pressure in the combustion chamber increases.



Complex 3D Motor



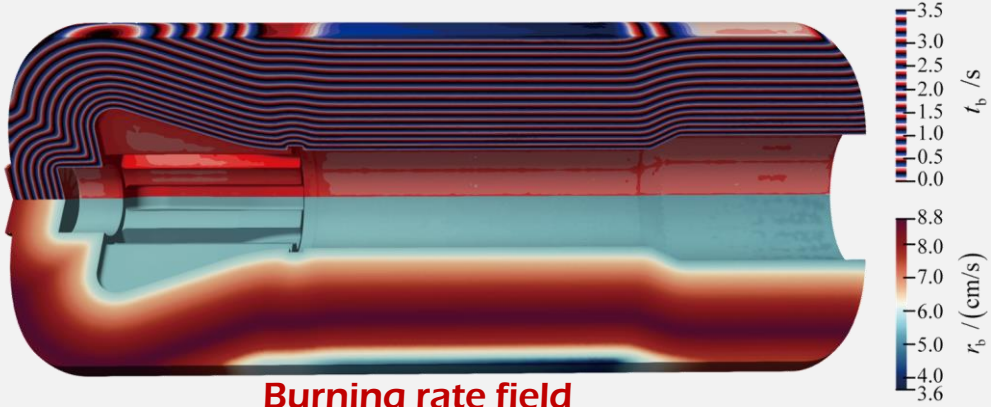
➢ The distribution of burning rate

- As time goes on, the burning surface aera first increases and then decreases, causing a unique burning rate field in the grain
- The combustion process can be visualized by the burned time field

➢ Motor with finocyl grain

- Finocyl grain is widely used in aerospace industry
- The slot can make it easier for designers to control the burning surface area (increase or decrease)
- Since the complexity of the geometry, it is better to generate polyhedron non-structural mesh

Burned time field

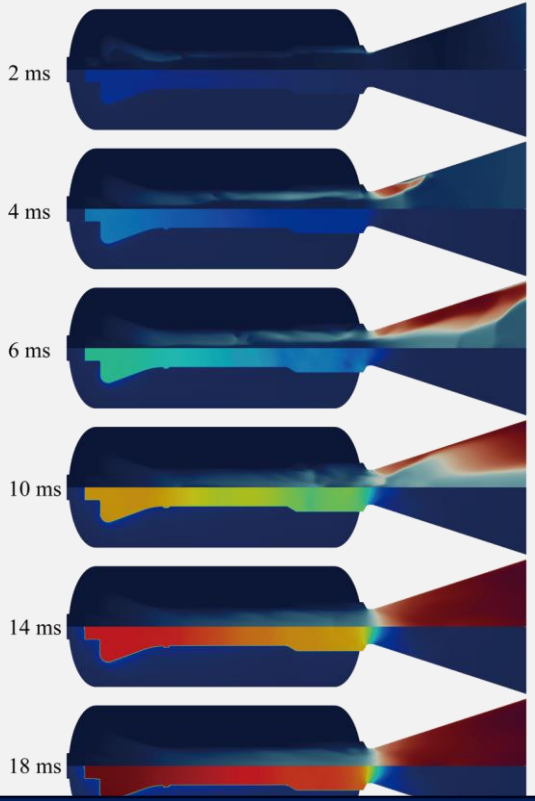


Burning rate field

Complex 3D Motor

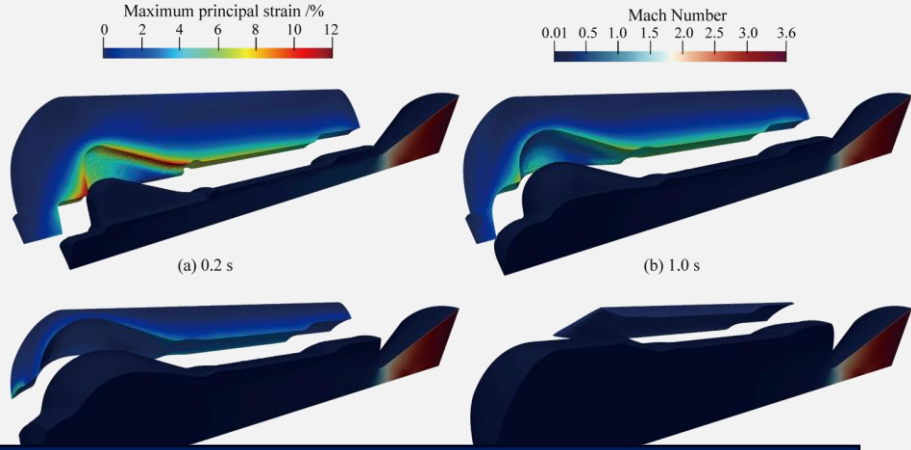
➢ Pressure establishment process

- It takes 18 ms to fill the cavity with burned gas
- After passing through the nozzle, the gas expands fully and accelerates
- There is no shock waves in the nozzle



➢ Whole working process

- It takes more than 3 seconds to finish the burn
- The maximum strain (principal strain) occurs at the edge of the slot, reaching 0.12

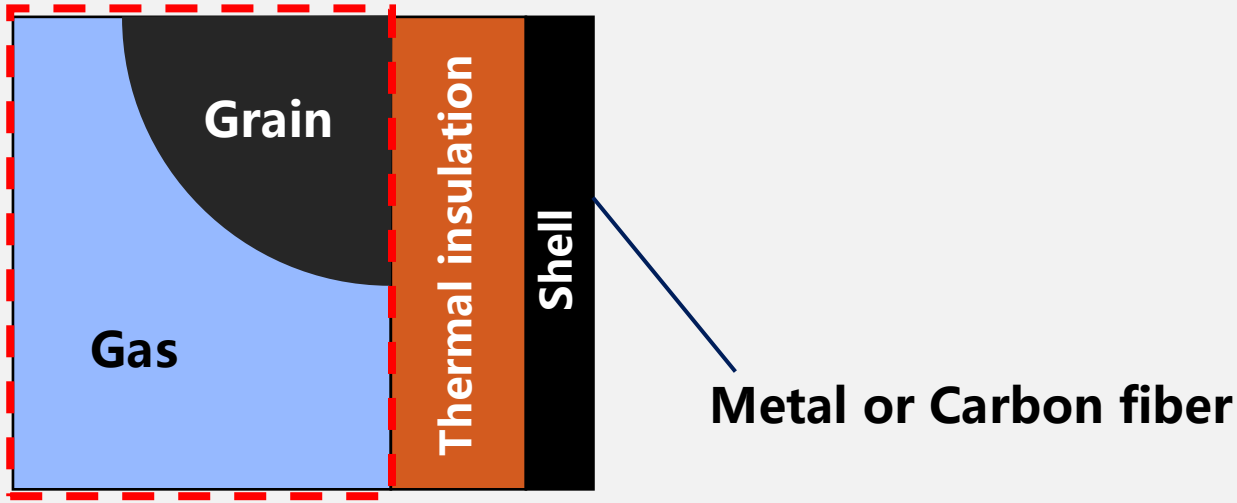


The SrmFOAM solver can handle the simulation of solid rocket motors with the burning surface regression

北京航空航天大學
BEIHANG UNIVERSITY

Future works

1 *Add thermal and stress calculations for combustion chamber and nozzle structures*



北京航空航天大學
BEIHANG UNIVERSITY

Future works

2 Add nozzle ablation model

3 Add ignition process and ignition model

$$\frac{\mathrm{d}T_s}{\mathrm{d}t} = \frac{4\alpha h^2 \left(T - T_S\right)^3}{3\lambda^2 \left(T_S - T_i\right)\left(2T - T_S - T_i\right)}$$

4 Add gas reaction in the flow
(Chemical equilibrium flow or detailed reaction mechanism)

5 Add particle phase in the flow

北京航空航天大學
BEIHANG UNIVERSITY

Conclusion

1 Principles of solid rocket motors (SRM)

1. Introduction
2. Internal ballistics and physical problems
3. Software available

2 Method of Multiphysics coupling

1. Objectives
2. Burning surface regression
3. Computational fluid dynamics
4. Computational solid mechanics

} Make them work on a static unstructured mesh

3 OpenFOAM Programming

1. Introduction of OpenFOAM
2. Introduction of our code

4 Result Case Show

1. The BATES motor, Attitude control motor, Nozzleless Motor, End-burning Motor…
2. Works in future

北京航空航天大學
BEIHANG UNIVERSITY

Main Reference

1. Heath MT, Dick WA. Virtual prototyping of solid propellent rockets. *Comput Sci Eng*. 2000; 2(2): 21-32.
2. 3. Rocstar multiphysics simulation suite [Internet]. Univer-sity of Illinois, Rocstar [updated 2020 Oct 6; cited 2023 Apr 15]. Avaliable from: https://github.com/IllinoisRocstar/Rocstar
3. Le Breton P, Ribereau D, Godfroy F, et al. Srm performance analysis by coupling bidimensional surface burnback and pressure field computations. *34th Joint Propulsion Conference and Exhibit*; 1998 July 13-15; Cleveland: AIAA; 1998.
4. Li WT, He YQ, Zhang YY, et al. Complex burning surface burn-back analysis and internal ballistic performance prediction of non-uniform grain. *Journal of Beijing University of Aeronautics and Astronautics*. 2022; [Chinese].
5. Openfoam [Internet]. The openfoam foundation [updated 2023 Jan 19; cited 2023 Apr 15]. Avaliable from: https://openfoam.org/.
6. Lorensen WE, Cline HE. Marching cubes: a high resolution 3d surface construction algorithm. *Computer Graphics*. 1987; 21(4): 163-169.
7. Wang B, Peng H, Chen YX. Effects of deformation on the burning behavior of solid propellants. *Int J Spray Combust*. 2016; 9(2): 116-126.
8. Brandyberry M. Uncertainty quantification in 3d rocket simulation. *42nd Joint Propulsion Conference & Exhibit*; 2006 Jul 9-12; Sacramento: AIAA; 2006.

Main Reference

9. Ballistic anomaly trends in subscale solid rocket motors, AIAA 82-1092
10. Burning rate enhancement phenomena in end-burning solid propellant grains, AIAA85-1435
11. Development and evaluation of the USAF ballistic test evaluation system for solid rocket propellants, AD 276-424
12. Effects of aluminum propellant loading and size distribution in BATES motors: A multiphysics computational analysis, AIAA 2005-3997
13. Nozzleless solid propellant rocket motors experimental and theoretical investigations, AIAA 84-1312
14. The history of the BATES motors at the Air Force Rocket Propulsion Laboratory, AIAA 98-3981
15. The relationship between solid propellant formulation variables and motor performance, AIAA 75-1199