

Reflected Shock Tunnel Analysis for Experimenters

Nick N. Gibbons,
The University of Queensland, Brisbane, Queensland 4072, Australia

February 15, 2024

What should I do to compute my test flow?

There's a long history of different codes for modelling reflected shock tunnels:

- ▶ 1960's: ESTC, NENZF (McIntosh @ ANU and Lordi et al. @ Cornell)
- ▶ 1990's: STN (PJ and Krek @ UQ),
- ▶ 2010's: ESTCj, NENZFr (PJ and many others @ UQ)
- ▶ Current: ESTCN, NENZF1d, Eilmer

▼ PUBLICATIONS/PRESENTATIONS

Journal Articles, Conference Papers and
Technical Reports

Theses

Presentations

▼ ALL THE CODES BY TYPE

2020

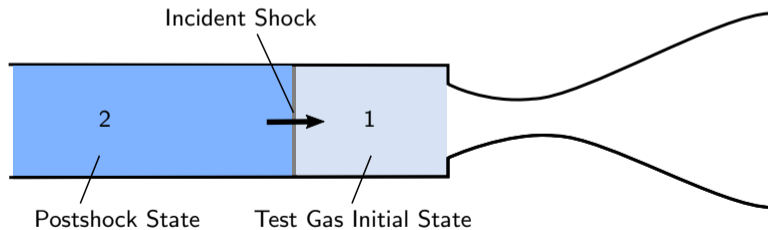

- [Calculation of Test Flow Conditions for Reflected-Shock Tunnels](#), 03 December 2020.

PAJ seminar to the Centre for Hypersonics research group on a couple of ways to estimate test flow conditions in reflected shock tunnels. The first uses Eilmer to do an axisymmetric simulation of the nozzle expansion and the second is a simpler quasi-one-dimensional calculation. Both make use of the finite-rate chemistry module.

What do these codes actually do?

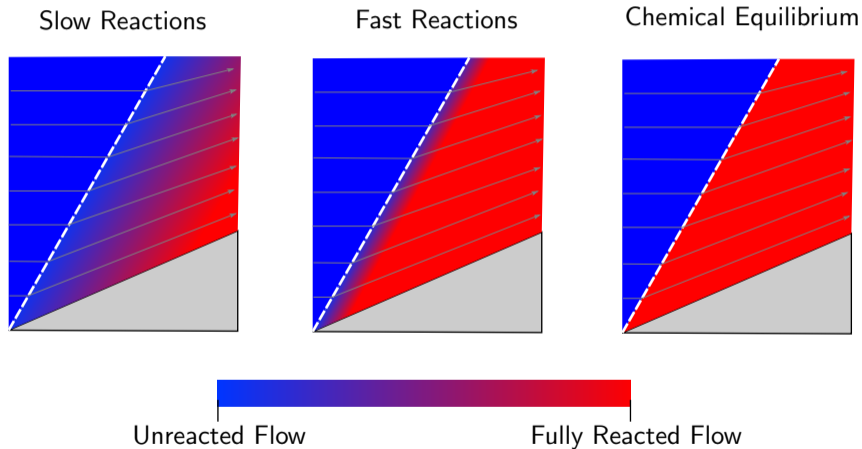
Step 1: Start with temperature/pressure of the shock tube gas and incident shock speed:

Step 1: Incident Shock Calculation



An aside about equilibrium shock calculations:

Most programs compute shock jumps using equilibrium chemistry:



An aside about equilibrium shock calculations:

Most codes used (and many still use) NASA CEA:

- ▶ Close source, old school FORTRAN 77, developed at NASA Lewis in the 80's-90's
- ▶ Inputs are text files that get read in and written out
- ▶ Interface is unpleasant and requires lots of work on our end

I've written my own code for this: github.com/uqngibbo/equilibrium-c

- ▶ Open source, modern C and Python
- ▶ Lightweight with minimal dependencies
- ▶ Nice interface to modern language via C bindings

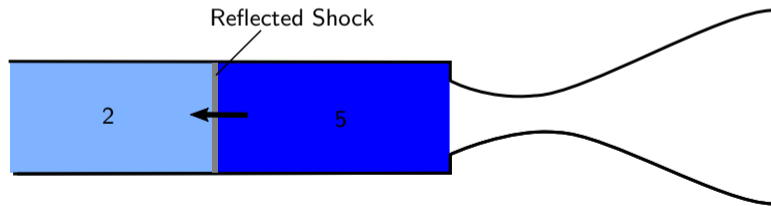
Shock Tunnel Calculations Step 2: Shock Reflection

Solve the shock jump problem assuming that postshock velocity is zero in the lab frame:

Step 1: Incident Shock Calculation

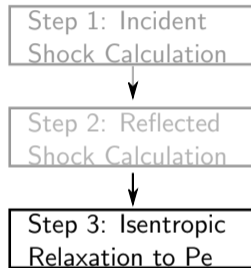
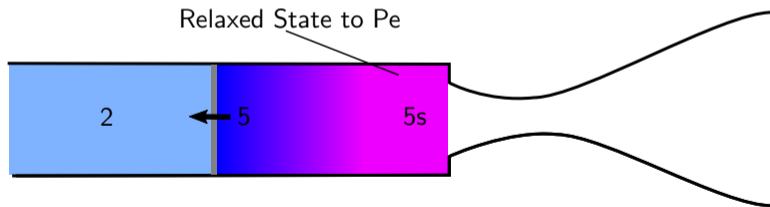


Step 2: Reflected Shock Calculation



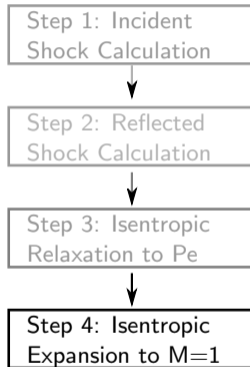
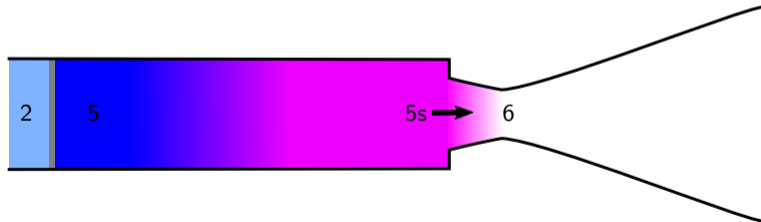
Shock Tunnel Calculations Step 3: Pressure Relaxation

Fudge the answer toward the measured stagnation pressure:



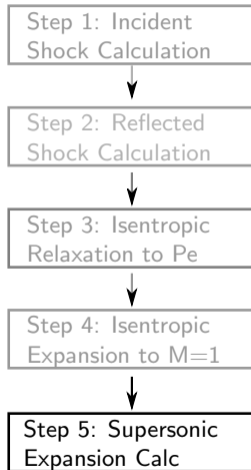
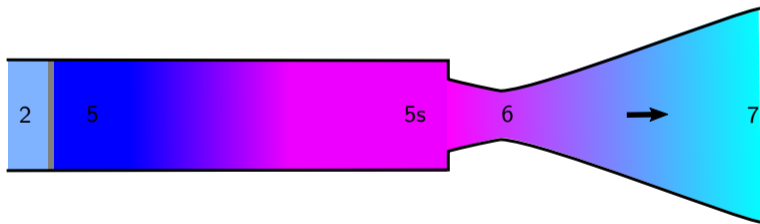
Shock Tunnel Calculations Step 4: Expansion to Mach 1

Compute up to the nozzle throat using isentropic expansion:



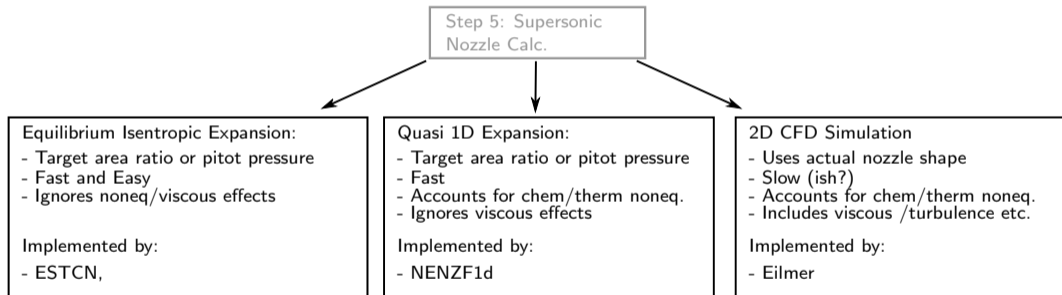
Shock Tunnel Calculations Step 5: Supersonic Expansion

The rest of the nozzle, expanding by area, or pressure, or something else...

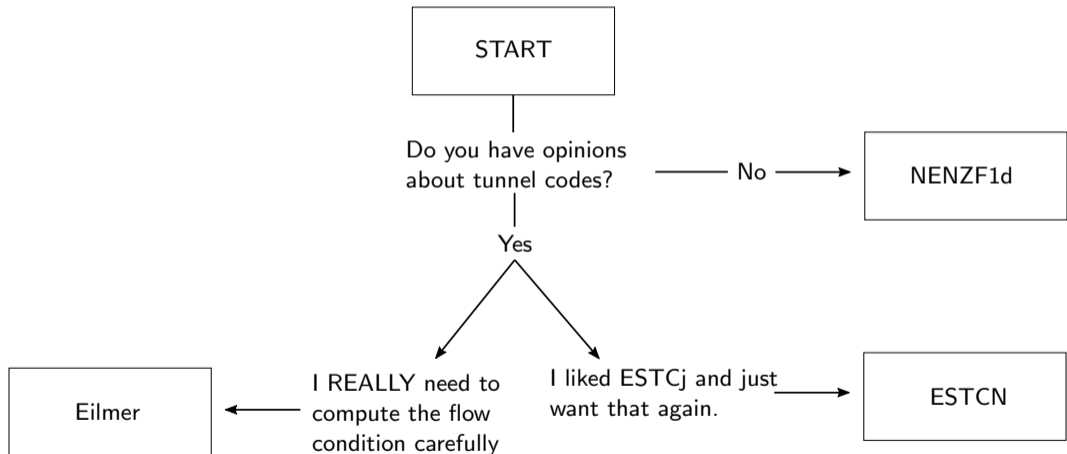


Shock Tunnel Calculations Step 5: Supersonic Expansion

Most of the options with different programs pertain to this part:



So which program should I use?



NENZF1d:

Quasi-1D nonequilibrium expansion of the nozzle region:

- ▶ Can stop at the end of the nozzle, or when a pitot/stag ratio reached
- ▶ Chemical Reactions and thermal nonequilibrium supported
- ▶ Works with CEA or equilibrium-c for the throat analysis

<https://gdtk.uqcloud.net/docs/nenzf1d/nenzf1d-manual-for-hugo>

GDTk [Docs](#) [People and Projects](#) [Blog](#) [News](#)

- > INTRODUCTION
- > GETTING STARTED
- > EILMER
- > L1D
- > PITOT
- > ESTCN
- ✓ NENZF1D

[NENZF1d Reference Manual](#)

- > PYTHON LIBRARIES
- > GRID UTILITIES
- > CFCFD (OLD) WEB SITE
- > PUBLICATIONS/PRESENTATIONS
- > ALL THE DOCS BY TYPE

NENZF1d Reference Manual

NENZF1D is a program for estimating flow conditions in reflected-shock tunnels, when the test gas reaches temperatures high enough for chemical reactions to occur and when nonequilibrium chemistry effects are expected to be important. The calculation proceeds in two parts. Assuming thermochemical-equilibrium, a state-to-state calculation is done for the shock-tunnel processing of the test gas until it reaches sonic conditions at the nozzle throat. Then a calculation with finite-rate chemical reactions is made of the supersonic flow of the gas through the nozzle expansion.

1. Getting started

The `nenzf1d` program is built upon the core gas model and flow modules of our gas-dynamics tool kit. General getting started notes can be found at <https://gdtk.uqcloud.net/docs/getting-started/prerequisites>. There, you will see how to get a copy of the source code, a list of what other software you will need to build and install the tool kit, and a collection of environment variables that need to be set.

To install the `nenzf1d` program, move to its source directory and use the `make` utility.

```
cd dgd/src/nenzf1d
make install
```

ESTCN:

Faithful (?) recreation of the old ESTCj code:

- ▶ Uses CEA and the Eilmer 4 gas models under the hood
- ▶ Called from command line in a familiar way
- ▶ Needs pitot-to-stag ratio or corrected area ratio

```
$ estcn --task=stn --gas=cea-lut-air.lua \  
      --T1=300 --p1=125.0e3 --Vs=2414 --pe=34.37e6 --ar=27.0
```

GDTk Docs People and Projects Blog News



> INTRODUCTION

> GETTING STARTED

> EILMER

> L1D

> PITOT

▼ ESTCN

ESTCN

[ESTCN Reference Manual](#)

ESTCN Reference Manual

ESTCN is a state-to-state calculation program for estimating flow conditions in reflected-shock tunnels. With a gas-model file already set up, it is intended to be used as a command-line tool for quick calculations of tunnel conditions.

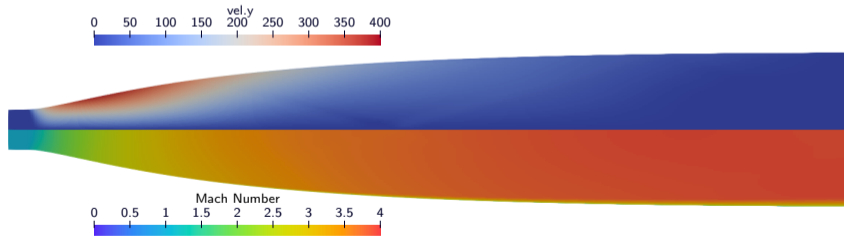
1. Getting started

The `estcn` program is built upon the core gas models and the Python library that is wrapped around that gas-model library. This is part of a larger gas-dynamics toolkit and general getting

Eilmer is an option!

Steady-state solver big improvements:

- ▶ 5-20 minutes on a workstation
- ▶ Can do reacting, turbulent, multi-temperature flow
- ▶ Examples take in shock speed/fill conditions in input scripts
- ▶ No fudging needed for viscous corrections



Comparison at Mach 8: Shot 11760

```
$ estcn --task=stnp --gas=cea-lut-air.lua --T1=300 --p1=194.0e3 \  
--Vs=2241.37 --pe=45.0e6 --pp_on_pe=7.011e-3
```

Estimated Exit State:

	Eilmer	NENZF1d		ESTCN	
v (m/s)	2926	2956	(+1.0%)	3005	(+2.7%)
T (K)	419	429	(+2.4%)	466	(+11.2%)
p (Pa)	4783	4738	(-0.9%)	4917	(+2.8%)
M	7.14	7.12	(-0.3%)	6.96	(-2.5%)

The Atomic Oxygen Problem

Almost all of the tunnel calcs we could find use 5 species air:

- ▶ N₂, O₂, N, O, NO
- ▶ Tristan discovered this this leads to artificially high amounts of O!

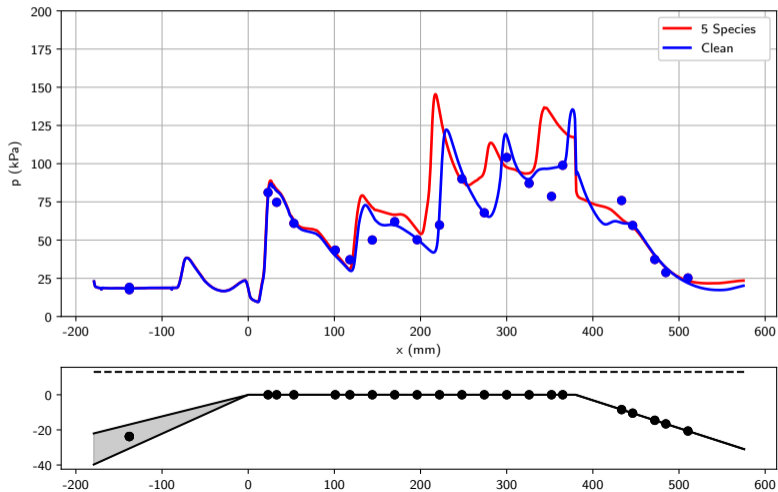
Shot 11760 outflow mass fractions with 5 species and 6 (+NO₂)

	5 Species	6 Species
N ₂	0.736	0.737
O ₂	0.196	0.199
N	4.06e-25	0.0
O	0.00107	4.24e-11
NO	0.0658	0.0619
NO ₂		0.000710

The Atomic Oxygen Problem

Eilmer steady-state calculation of Tristan's PhD experiment:

- ▶ 6.8M cells, 7 species reacting, SA turbulent, 4.5 hours on 1024 cores (Setonix)
- ▶ Compare 0.001 mass fraction of O vs zero O



The Atomic Oxygen Problem

- ▶ Right now lots of the nenzf1d examples still use 5 species air (sorry)
- ▶ We're experimenting with a 7 species NOX scheme from UCSD that looks good
- ▶ Once I'm happy all of the scripts will be ported to fix this (sorry again)

Nitrogen Chemistry

Date	FlameMaster	Chemkin	Thermo Data	Transport Data	Documentation (With References)	Update Notes
2018-07-23	FLM_2018-07-23	CK_2018-07-23	THERM_2016-08-15	TRANS_2016-08-15	Mechanism_2018-07-23	Notes_2018-07-23
2004-12-09 (2)	FLM_2004-12-09	CK_2004-12-09	THERM_2005-03-10	TRANS_2002-10-01	Mechanism_2004-12-09	Notes_2004-12-09
2004-12-09 (1)	FLM_2004-12-09	CK_2004-12-09	THERM_2002-10-01	TRANS_2002-10-01	Mechanism_2004-12-09	Notes_2004-12-09
2003-08-12	FLM_2003-08-12	CK_2003-08-12	THERM_2002-10-01	TRANS_2002-10-01	Mechanism_2003-08-12	Notes_2003-08-12

Figure 1: web.eng.ucsd.edu/mae/groups/combustion/mechanism.html

Coming soon: Standalone Python Script

One of the examples in `equilibrium-c` is just the same thing as ESTCN

- ▶ Cross-platform (thanks Oliver Street!), lightweight, easy to install
- ▶ Ideally we build it into a properly maintained tunnel toolkit
- ▶ Zero dependencies and integrated with PyShot-like program

```
uqngibbo@melchior:~/.../equilibrium-c/examples$ python3 shocktube.py
```

```
s7
```

```
p: 32712.207 Pa T: 830.250 K rho: 0.137 kg/m3 v: 2166.979 m/s  
h: 552788.454 J/kg s: 8248.303 J/kg/K Mmix: 0.028965 kg/mol M: 3.819564  
N2:0.7551779, Ar:0.0129160, CO2:0.0004847, O2:0.2314180, NO:0.0000034
```

The Future

Can we do any better?

- ▶ The tools in this talk are for production post-processing
- ▶ Methods haven't changed much in 50 years
- ▶ What if I want to modify the condition or tunnel itself?

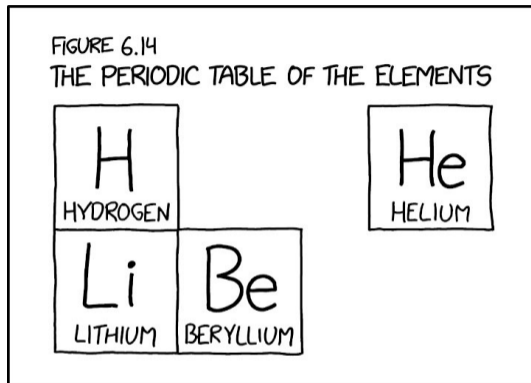
More predictive analysis is possible with different tools:

- ▶ I1d: Lagrangian quasi-1D for unsteady flow analysis
- ▶ Eilmer: Kyle's (not yet available) BDF2 method for time-accurate implicit steps
- ▶ PITOT: Chris James's code can apparently work for RST's?

Final Thoughts

- ▶ Mass fractions computed by ESTCN are not accurate at all
- ▶ Don't use 5 species air if you are doing combustion experiments!
- ▶ Accounting for shock attenuation is hard but probably worth trying
- ▶ We should look at using L1d for shot analysis, at least to benchmark accuracy

Thanks!



YOU CAN SPOT AN OUTDATED SCIENCE TEXTBOOK BY CHECKING THE BOTTOM OF THE PERIODIC TABLE FOR MISSING ELEMENTS. FOR EXAMPLE, MINE WAS PUBLISHED HALF AN HOUR AFTER THE BIG BANG.