NENZF-r: Non-Equilibrium Nozzle Flow, Reloaded. A User Guide.

The University of Queensland
School of Mechanical & Mining Engineering
Research Report Number 2011/??

L. Doherty, F. Zander, P. A. Jacobs,[*]
R. J. Gollan, W.Y.K. Chan and R. M. Kirchhartz
Centre for Hypersonics, The University of Queensland, Brisbane, Australia.

June 4, 2012

Abstract

This report blah blah blah

[*]Queensland Geothermal Energy Centre of Excellence, The University of Queensland, Brisbane, Australia.

Contents

1 Introduction 3

2 Overview 4

3 NENZFr 5
- 3.1 Useage Tips 5
- 3.2 To do 5
- 3.3 Method 6
- 3.4 Inputs 6
- 3.5 Output Files 8
- 3.6 Gas Models and Reaction Schemes 8
  - 3.6.1 Air 8

4 NENZFr Perturbed 11
- 4.1 Useage Examples 11
- 4.2 Useage Notes 11
- 4.3 Method 12
  - 4.3.1 Perturbing for Sensitivity Calculation 12
  - 4.3.2 Perturbing for Response Surface 12
- 4.4 Inputs 12
- 4.5 Outputs 12

5 NENZFr Sensitivity 14
- 5.1 Useage Examples 14
- 5.2 Method 14
  - 5.2.1 3-point derivative 14
  - 5.2.2 5-point derivative 15
- 5.3 Inputs 16
- 5.4 Outputs 16

6 NENZFr Response Surface 17
- 6.1 Useage Examples 17
- 6.2 Method 18
  - 6.2.1 2nd-order Surface 18
  - 6.2.2 Radial-Basis Function Surface 19
- 6.3 Inputs 19
- 6.4 Output Files 19

7 NENZFr Quasi-Transient 20
- 7.1 Method 20
- 7.2 Inputs 20
- 7.3 Outputs 20

1 Introduction

For many years the calculation of the free-stream test flow properties within Shock Tunnel Facilities has relied upon a suite of codes, including, but not limited to ESTC[1][1], ESTCj[2], STN[3][] and NENZF[4][2]. Each of these codes serves a different purpose. ESTC, written in the late 1960's in Fortran IV, was used to calculate to nozzle supply conditions in reflected shock tunnels assuming thermodynamic equilibrium.

Difficulty in maintaining the thermodynamic database used by ESTC led Dr. Peter Jacobs to update this code in the mid 1990's. The resulting program, ESTC-junior, was written in Python and relied on calls to NASA's CEA[3] program for the thermodynamic properties. NENZF, another product of the 1960's, uses nozzle supply conditions and an input nozzle geometry to completes a quasi-one-dimensional expansion assuming either frozen, equilibrium or non-equilibrium gas.

The final code, STN, is another Fortran based program that was developed in the mid 1990's. Implemented using thermodynamic curve fits for equilibrium air only, this code was written in order to provide experimenters with freestream properties for low-enthalpy conditions ($<\approx$3.5MJ/kg) for which other codes would crash.

A desire to consolidate the programs, and in particular to update and move away from NENZF, has existed within the Centre for Hypersonics for some time. Much like the original ESTC, the thermodynamic properties, reaction rates and reaction schemes implemented within NENZF are not easily maintained. Furthermore, difficulty in obtaining a converged solution for particular nozzle geometries over particular enthalpy ranges has been a consistent issue with NENZF.

This example is a proof-of-concept showing that the aging ESTC and NENZF codes can be replaced by something with more up-to-date chemistry and with a better approximation to the multi-dimensional expansion that processes the test gas.

The request was for a calculation yielding single numbers for each flow variable and a run time that was not too long.

This has been made possible with the recent implementation of a space-marching algorithm within Eilmer3[]

[1]**E**quilibrium-**S**hock-**T**ube-**C**alculation
[2]**E**quilibrium-**S**hock-**T**ube-**C**alculation-**j**unior
[3]**S**hock-**T**ube-**N**ozzle
[4]**N**on-**E**quilibrium-**N**ozzle-**F**low

2 Overview

3 NENZFr

NENZF-r is comprised of a set of Python scripts and template files that coordinate the running of a space-marched nozzle simulation using Eilmer3. The following sections provide an overview of the implementation of NENZF-r and its capabilites. Though based on Eilmer3, minimal details concerning Eilmer3 are provided in this report. Readers are instead referred to the reports by [] for further information on Eilmer3 and on the space-marching algorithm. Table 1, given below, summarises the scripts that comprise NENZF-r. A copy of each of these files in it's current form may be found in Appendices **??** - **??**.

3.1 Useage Tips

- Aim to have 10 cells per block in the axial direction for both block-sequencing and block-marching calculations. Having less for block-sequencing influences the final result[5]. For block-marching calculations 10 cells per block in the radial direction is also recommended.
- When running simulations on Barrine we are currently limited to 1 node (8 cpus in total). Thus for block-marched calculations `--nbj` is limited to 4.
- The air gas model **should not** be used for non-equlibrium calculations. See Section 3.6.
- The Mach 10 nozzle is a pain. Don't use it. If you must, then don't set `BLTrans` to 50 mm. For some reason this value causes Eilmer3 to crash. Other values (100 mm, 150 mm) including a fully turbulent nozzle (`BLTrans`=0.0) seem to be ok. If you can find out why, let me know.
- The default values for all nenzfr inputs have been set up for a Mach 10, chemical equilibrium, laminar nozzle calculation. If you are running a different nozzle, be sure to check the grid parameters (`nni`, `nnj`, nbi etc.) and the maximum simulation time (`max-time`).
- The building of the equilibrium gas LUT takes a long time. Try to get a copy from someone. If you put it in the directory from which you will be calling *nenzfr* (or the other related codes) then *nenzfr* won't try to build it.

3.2 To do

- The CEA LUTs used for equilibrium nozzle calculations include ions. The Gas objects defined by *make_gas_from_name()* within *cea2_gas.py* do not include ions. These Gas objects are used by *estcj* during the calculation of the nozzle supply condition. When no-ions LUTs were created, Wilson found that they caused NENZFr to crash for some reason. No further work has been done at this time (04-06-2012).
- When completing a sensitivity calculation for the Mach 4, equilibrium, laminar test case, it was found that the inflow turbulence intensity had non-zero sensitivity. This shouldn't be the case for a laminar calculation. Wilson has looked at this and found

[5]We assume that the same would be true for block-marching. This has yet to be checked.

that the error originates from the inflow plane. No further work has been done at this time (04-06-2012).

- Once the inflow turbulence intensity issue has been sorted this input should not be perturbed in *nenzf_perturbed* for laminar nozzles.

3.3 Method

NENZF-r has been designed specifically to estimate the test flow conditions produced in reflected shock tunnels. Referring to Figure **??**, given the intial conditions of the gas in the shock tube (state 1), the shock speed and nozzle supply pressure, the global algorithm used by NENZF-r is:

1. compute the gas conditions behind the incident shock (state 2);
2. compute the gas conditions after shock reflection (state 5);
3. complete an isentropic relaxation to the measured nozzle supply pressure (state 5s);
4. complete an isentropic relaxation to the nozzle throat (state 6);
5. from the throat, complete a 2D axis-symmetric expansion of the gas to the nozzle exit (state 7).

Steps 1-4 are completed by *estcj.py* assuming an equilibrium gas and using 1D shock relations. Calls to NASA's CEA [3] program are used to determine the thermodynamic properties. Step 5, the nozzle expansion, is completed using Eilmer3 assuming either an equilibrium, non-equilibrium or frozen gas. Viscosity and turbulence may be considered depending on the Eilmer3 simulation control data specified in *nozzle.input.template*.

The top-level coordinating script, *nenzfr.py*, writes an Eilmer3 input file based on *nozzle.input.template*, calls Eilmer3 and then completes the necessary post-processing to determine the averaged properties at the nozzle exit. The following sections discuss the inputs, output files produced and gas models/reaction schemes currently implemented.

3.4 Inputs

NENZF-r takes a number of required and optional input arguments as detailed below.

`--help` Displays help information including listing the available input arguments. No other inputs are relevant.

Necessary inputs:

`--T1` Shock tube filling temperature (K)

`--p1` Shock tube filling pressure (Pa)

`--Vs` Incident shock speed (m/s)

`--pe` (Measured) Nozzle supply pressure (Pa)

Optional inputs:

Table 1: NENZF-r scripts

Core Files	
nenzfr.py	Top most coordinating script.
nozzle.input.template	Template Eilmer3 input script. This is the core of NENZF-r.
Auxilary Files	
nenzfr_compute_viscous_data.py	Script used to calculate the viscous data, including y^+, at the wall.
nenzfr_sensitivity_and_lut.py	Coordinating script that conducts a sensitivity analysis for NENZF-r and builds a corresponding look-up-table for the nozzle exit properties. (YET TO BE COMPLETED)
setup_lut.sh	A simple shell script that builds the required equilibrium gas look-up-tables.
run.sh	Shell script to submit a NENZF-r job on the hpcu cluster *barrine*.
estcj.py	Equilibrium Shock Tube Calculation, junior. This Python script completes the reflected shock calculation. Though apart of the cfcfd2/Eilmer3 suite of programs, it has been included here for completeness.
Nozzle Profiles	
contour-t4-m10.data	T4 Mach 10 nozzle profile.
contour-t4-m6.data	T4 Mach 6 nozzle profile.
contour-t4-m4.data	T4 Mach 4 nozzle profile.
Reaction Schemes	
gupta_etal_air_reactions.lua	Reaction scheme for 5-species air model.
nitrogen-5sp-6r.lua	Reaction scheme for nitrogen.

`--job` Base name for solution files. Default: nozzle

`--gas` Gas that the shock tube has been filled with. Default: air. Options: air, air5species, n2

`--chem` Chemistry model to use in the nozzle expansion. Default: eq. Options: eq, neq, frz, frz2

`--cfile` Filename for the contour defining the shape of the nozzle. Default: contour-t4-m4.data

`--gfile` Filename for a Pointwise plot3d grid for the nozzle. This option over-rides *–cfile* if both are given.

`--area` Exit area for the nozzle. Only used by *estcj* which determines the nozzle exit properties (state 7) via an isentropic expansion. Default: 27.0

`--exitfile` Filename for the nozzle exit slice data. Default: nozzle-exit.data

`--just-stats` Option allowing the code to skip the detailed calculation and just retrive the exit-flow statistics

Several example invocations for NENZF-r are as follows.

- Mimimal. This completes an equilibrium calculation using air for the Mach 10 nozzle assuming that the nozzle is fully laminar.
  ```
  ./nenzfr.py --T1=300 --p1=160.0e3 --Vs=2707 --pe=38.0e6
  ```
- Nitrogen as the test gas:
  ```
  ./nenzfr.py --T1=300 --p1=160.0e3 --Vs=2707 --pe=38.0e6 --gas=n2
  ```
- A non-equilibrium solution for the Mach 10 nozzle using the 5-species air gas model:
  ```
  ./nenzfr.py --gas=air5species --T1=300.0 --p1=160.0e3 --Vs=2300.0
  --pe=43.0e6 --cfile=contour-t4-m10.data --area=1581.165 --chem=neq
  ```

3.5 Output Files

Providing that NENZF-r has been properly invoked and that the Eilmer3 nozzle simulation has not crashed, users can expect the files detailed in Table 2 to be produced. Note that these files are in addition to those produced by Eilmer3.

Table 2: NENZF-r output files

Filename	Default Filename	Description
job-estcj.dat	nozzle-estcj.dat	Output file detailing the results of the estcj calculation.
exitfile	nozzle-exit.data	Flow properties in the exit plane.
job-centerline.data	nozzle-centreline.data	Flow properties along the centerline.
job-exit.stats	nozzle-exit.stats	The integrated nozzle exit flow properties.
job-viscous.data	nozzle-viscous.dat	Calculated wall shear stress (τ_w) and y^+.

Currently the integrated nozzle exit flow properties are calculated by taking an area-weighted average over the core of the nozzle using the data contained in *nozzle-exit.data*. The desired core radius can be specified as a fraction of the nozzle exit radius using the option `--CoreRadiusFraction`. Quite arbitarily, the default value has been taken to be $\frac{2}{3}$.

3.6 Gas Models and Reaction Schemes

3.6.1 Air

This gas model should only be used when completing equilibrium or frozen gas calculations. If chemical non-equilibrium is specified the code swaps to the 5-species air model for the nozzle calculation. The reason for doing this is simply that the CEA equilibrium air model considers many more species than would be practical[6] to include in a

[6]from computational-expense considerations

non-equilibrium nozzle flow calculation. The species present at the throat would also be dependent on the condition and finding a suitable reaction scheme is difficult.

The shift from the complete air model to the 5-species model occurs at the throat. The current implementation finds the equilibrium temperature required to conserve mass, momentum and energy with the change in gas composition assuming that the static pressure and velocity are constant across the throat. The theory is outlined below.

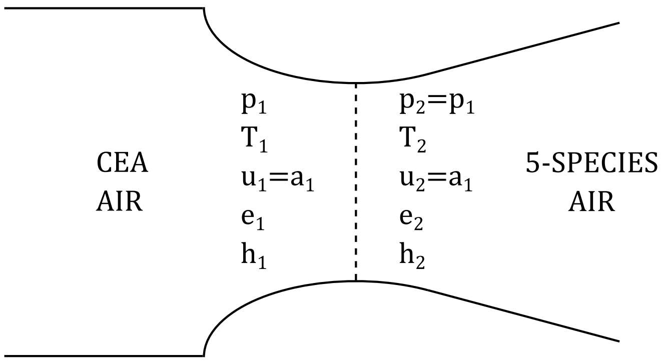


Figure 1

Using conservation of mass, momentum and energy flux and with reference to Figure 1 we may write:

$$\rho_1 a_1 = \rho_2 a_2 \tag{1}$$
$$p_1 + \rho_1 a_1^2 = p_2 + \rho_2 a_2^2 \tag{2}$$
$$h_1 + \frac{a_1^2}{2} = h_2 + \frac{a_2^2}{2} \tag{3}$$

where the fact that $u_1 = a_1$ and $u_2 = a_2$ has been used. To solve the above equations under a change in gas composition an additional assumption is required for one of the state variables. Noting that in a physical nozzle a change in static pressure across the throat would cause a wave to be driven either upstream or downstream, we make the assumption that $p_2 = p_1$. The momentum flux, Equation (2), thus reduces to:

$$\rho_1 a_1^2 = \rho_2 a_2^2 \tag{4}$$
$$\therefore a_1 = a_2 \tag{5}$$

using Equation (1). Consequently Equation (3) becomes:

$$h_1 = h_2$$
$$\therefore h_1 = e_2 + \frac{p_2}{\rho_2}$$
$$\therefore h_1 = e_2 + R_2 T_2 \tag{6}$$

where the definitions of static enthalpy and the ideal gas equation of state have been used. Since the gas is in equilibrium and p_2 is know then $e_2 = e_2(T_2)$ and $R_2 = R_2(T_2)$. Thus we may rewrite Equation (6) as:

$$f(T_2) = 0 = e_2 + R_2 T_2 - h_1 \tag{7}$$

Equation (7) can be solved numerically to deterime the temperature T_2 of the 5-species air gas composition that conserves mass, momentum and energy.

Frozen Calculation: There are two frozen calculation options available for this gas model. For `chem=frz` the CEA air species present at the throat are used for the frozen nozzle calculation. For `chem=frz2`, the code changes to the 5-species air model as described above and then uses these species for a frozen nozzle calculation. Really, you should just use the 5-species air model.

4 NENZFr Perturbed

This code controls and coordinates the running of a series of *nenzfr* calculations that are perturbations around some given nominal condition. The perturbed results may be used to either determine the sensitivity of each freestream property to the various inputs or to build a response surface for use during an experimental campaign. Refer to Section 5 and Section 6 for further details on the codes that complete these different analyses.

4.1 Useage Examples

- List all inputs and their default values:
  ```
  nenzfr_perturbed.py --help
  ```

- Perturb the given nominal condition using default perturbation Δ's on a standard Linux machine. In this case the inputs are identical to a typical *nenzfr* call. The below example runs a turbulent calculation for the Mach 4 nozzle in non-equilibrium mode. Turbulence is switched on 100mm downstream of throat:
  ```
  nenzfr_perturbed.py --gas=air5species --chem=neq --area=27.0
  --cfile='Bezier-control-pts-t4-m4.data' --block-marching
  --p1=170.0e3 --T1=300.0 --Vs=2250.0 --pe=25.0e6 --BLTrans=0.100
  --nni=600 --nnj=50 --nbi=60 --nbj=5 --bx=1.05 --by=1.002
  --max-time=0.003
  ```

- Run the simulations on the UQ Barrine cluster. In this case we would use the following options:
  ```
  ...--runCMD=qsub --cluster=Barrine
  ```

- Perturb **only** V_s and p_e with the aim of building a response surface for the nominal condition. Set the perturbations for V_s and p_e to $\pm 6\%$ and $\pm 14\%$ respectively:
  ```
  ...--create-RSA --Vs='[2250.0,6]' --pe='[25.0e6,14]'
  ```

- Give absolute values for the perturbation Δ and specify that 5 levels are to be used:
  ```
  ...--levels=5 --perturb=abs --p1='[170.e3 2.e3]'
  --Vs='[2250.  50.  -25.]'  --pe='[25.e6 2.e6 -4.e6 5.e6 -7.e6]'
  ```
  In this case the values used for each input will be:

  $$p_1 \in \{166.0e3, 168.0e3, 170.0e3, 172.0e3, 174.0e3\}$$
  $$V_s \in \{2200, 2225, 2250, 2300, 2350\}$$
  $$p_e \in \{18.e6, 21.e6, 25.e6, 27.e6, 30.e6\}$$

  Similar formatting may also be used for `--perturb=rel`.

4.2 Useage Notes

- If the nozzle is either fully turbulent or fully laminar then `BLTrans` will not be perturbed for a sensitivity calculation.

- For a fully laminar nozzle the Turbulent-to-Laminar viscosity ratio (`TurbVisRatio`) is also not perturbed. The inflow turbulence intensity (`TurbIntensity`) is perturbed but only because there is some underlying issue that needs sorting. See Section 3.2.

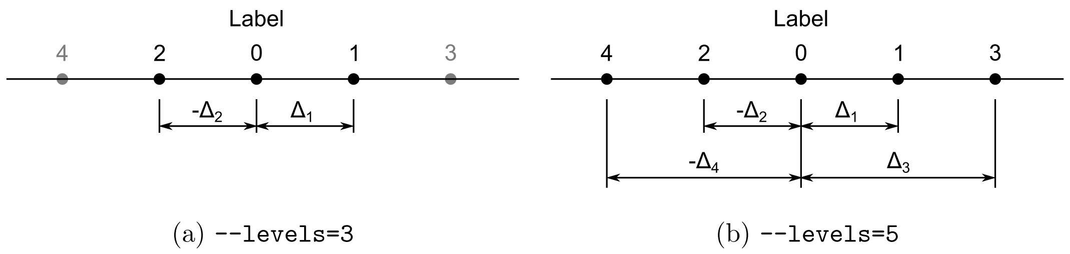
4.3 Method

4.3.1 Perturbing for Sensitivity Calculation

When running *nenzfr_perturbed* in preparation for completing a sensitivity calculation each of the inputs listed in Table 3 are perturbed. The default perturbations are also given. The `--levels` option dictates how many cases will be ran for each perturbed variable as shown in Figure 2.

Table 3: NENZFr Input Parameters Perturbed for Sensitivity Calculations

Input	Default Perturbation
p_1	2.5%
T_1	2.5%
V_s	2.5%
p_e	2.5%
T_{wall}	2.5%
x_{tr}	2.5%
I_{turb}	2.5%
μ_{turb}/μ_{lam}	2.5%
r_{core}/r_{exit}	2.5%



(a) `--levels=3` (b) `--levels=5`

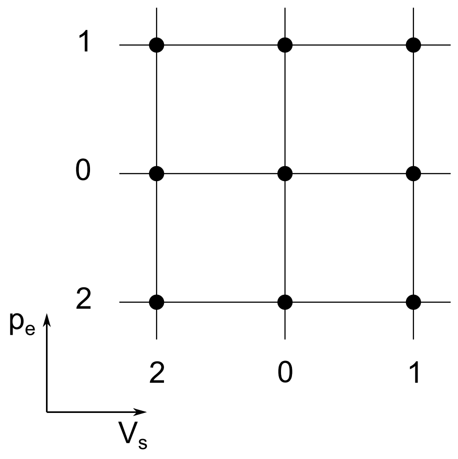
Figure 2: Perturbation cases considered for a sensitivity analysis. Label 0 represents the nominal values, labels 1-4 are the perturbations.

4.3.2 Perturbing for Response Surface

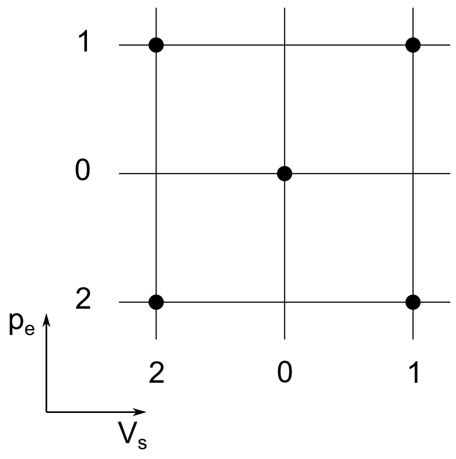
When the `--create-RSA` option is selected only V_s and p_e are perturbed. The reasons for this are explained at the start of Section 6. Any perturbation Δ's specified for the other inputs are simply ignored in the code. Figure 3 summarises the cases that are used when creating a response surface.

4.4 Inputs

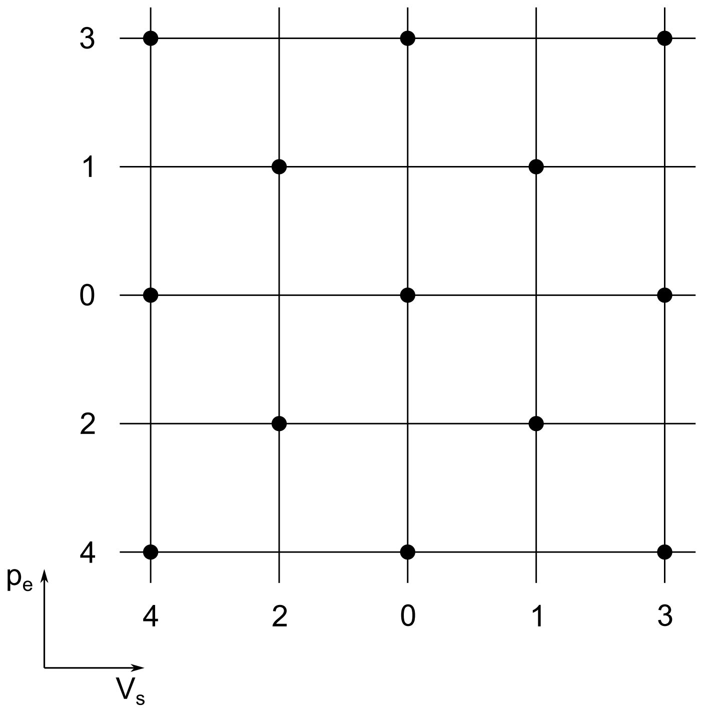
4.5 Outputs



(a) `--levels=3`, 9 cases in total



(b) `--levels=3-reduced`, 5 cases in total



(c) `--levels=5`, 13 cases in total

Figure 3: Perturbation cases used for calculation of a Response Surface. Case 00 is the nominal condition.

5 NENZFr Sensitivity

This code is used to analyse a set of NENZFr perturbation cases and calculate the uncertainties in each freestream property. The calculation methodology implemented is based on that described in [4].

5.1 Useage Examples

nenzfr_sensitivity should be called from the same directory in which *nenzfr_perturbed* was ran. Below are some typical invocations.

- List all inputs and their default values:
  `nenzfr_sensitivity.py --help`
- Calculate the sensitivities using default uncertainties and a three point derivative:
  `nenzfr_sensitivity.py --run-defaults`
- Calculate sensitivities using a 5-point derivative:
  `nenzfr_sensitivity.py --levels=5`
- Calculate sensitivities using different (relative) uncertainties for V_s, p_e and p_1:
  `nenzfr_sensitivity.py --XVs=0.025 --Xpe=0.032 --Xp1=0.02`

5.2 Method

The default uncertainties for the measured inputs were taken from [5] while the defaults for the remaining inputs were guestimated by the authors. The values are summarised in Table 4.

Table 4: Default Uncertainties for NENZFr Input Parameters

Input	Relative Uncertainty
p_1	3.25%
T_1	2%
V_s	5%
p_e	2.5%
T_{wall}	4%
x_{tr}	100%
I_{turb}	80%
μ_{turb}/μ_{lam}	100%
r_{core}/r_{exit}	5%

5.2.1 3-point derivative

When calculating the derivative at x_0, we make no assumption concering the spacing of x_+ and x_- relative to x_0. With reference to Figure 4 then using Taylor series expansion around x_0 we may write the following equations:

$$f_+ = f_0 + f_0' h_+ + \frac{1}{2} f_0'' h_+^2 + O(h_+^3) \tag{8}$$

$$f_- = f_0 + f_0' h_- + \frac{1}{2} f_0'' h_-^2 + O(h_-^3) \tag{9}$$

where $h_i = x_i - x_0$, $f_i = f(x_i)$, $f_i' = f'(x_i)$ and $f_i'' = f''(x_i)$.

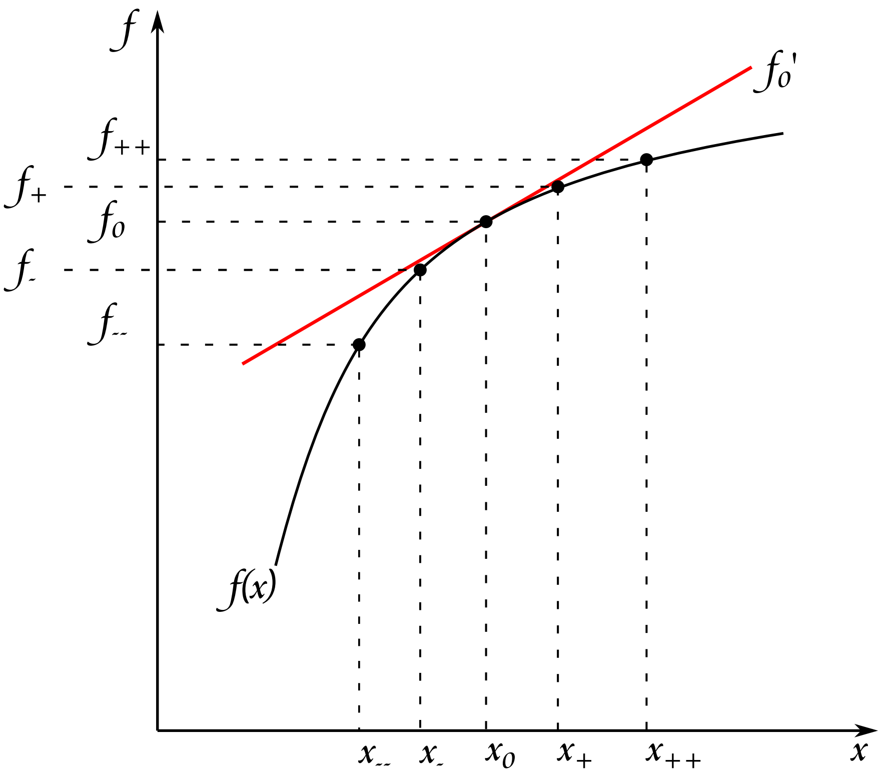


Figure 4: Calculation of the gradient using either 3 or 5 points.

We want to combine these two equations such that the second order terms cancel. Therefore, multiply Equation (9) by h_+/h_- and subtract the result from Equation (8) to give:

$$f_+ - f_- \frac{h_+}{h_-} = \left[1 - \left(\frac{h_+}{h_-}\right)^2\right] f_0 + \left[h_+ - \frac{h_+^2}{h_-}\right] f_0' + O(h_+^3) - O(h_-^2 h_+)$$

Rearranging the above we arrive at the following expression for the derivative at x_0:

$$f_0' = \left(-\frac{h_-}{x_+ - x_-}\right) . \frac{f_+ - f_0}{h_+} + \left(\frac{h_+}{x_+ - x_-}\right) \frac{f_- - f_0}{h_-} \tag{10}$$

where the terms in brackets are weightings for the forward and backward two-point derivatives. The truncation error for this estimate of the derivative is: $O\left(\max(h_+^2, h_-^2) . \frac{h_-}{h_+ - h_-}\right)$

5.2.2 5-point derivative

For the calculation of the derivative at x_0 using five points, we write the following Taylor series expansions:

$$f_{++} = f_0 + f_0' h_{++} + \frac{1}{2} f_0'' h_{++}^2 + \frac{1}{6} f_0''' h_{++}^3 + O(h_{++}^4) \tag{11}$$

$$f_{--} = f_0 + f_0' h_{--} + \frac{1}{2} f_0'' h_{--}^2 + \frac{1}{6} f_0''' h_{--}^3 + O(h_{--}^4) \tag{12}$$

$$f_{+} = f_0 + f_0' h_{+} + \frac{1}{2} f_0'' h_{+}^2 + \frac{1}{6} f_0''' h_{+}^3 + O(h_{+}^4) \tag{13}$$

$$f_{-} = f_0 + f_0' h_{-} + \frac{1}{2} f_0'' h_{-}^2 + \frac{1}{6} f_0''' h_{-}^3 + O(h_{-}^4) \tag{14}$$

Following the procedure used in the previous section we combine Equation (11) and Equation (12) to eliminate the second order terms. Thus, multiply Equation (12) by h_{++}^2/h_{--}^2 and subtract from Equation (11) to give:

$$f_{++} - f_{--}\frac{h_{++}^2}{h_{--}^2} = f_0 \left[1 - \frac{h_{++}^2}{h_{--}^2}\right] + f_0' h_{++} \left[1 - \frac{h_{++}}{h_{--}}\right] + \frac{1}{6} f_0''' h_{++}^2 \left(x_{++} - x_{--}\right) \tag{15}$$

In a similar way combine Equation (13) and Equation (14) to give:

$$f_{+} - f_{-}\frac{h_{+}^2}{h_{-}^2} = f_0 \left[1 - \frac{h_{+}^2}{h_{-}^2}\right] + f_0' h_{+} \left[1 - \frac{h_{+}}{h_{-}}\right] + \frac{1}{6} f_0''' h_{+}^2 \left(x_{+} - x_{-}\right) \tag{16}$$

The goal is to now combine Equations (15) and (16) such that the terms involving f_0''' are eliminate. We therefore multiply Equation (16) by:

$$\frac{h_{++}^2}{h_{+}^2} \frac{x_{++} - x_{--}}{x_{+} - x_{-}} \tag{17}$$

which gives:

$$\begin{aligned}\left[\frac{f_{+}}{h_{+}^2} - \frac{f_{-}}{h_{-}^2}\right] h_{++}^2 \left(\frac{x_{++} - x_{--}}{x_{+} - x_{-}}\right) &= f_0 \left[\frac{1}{h_{+}^2} - \frac{1}{h_{-}^2}\right] h_{++}^2 \left(\frac{x_{++} - x_{--}}{x_{+} - x_{-}}\right) \\ &\quad + f_0' \left[\frac{1}{h_{+}} - \frac{1}{h_{-}}\right] h_{++}^2 \left(\frac{x_{++} - x_{--}}{x_{+} - x_{-}}\right) \\ &\quad + \frac{1}{6} f_0''' h_{++}^2 \left(x_{++} - x_{--}\right)\end{aligned} \tag{18}$$

We now subtract Equation (18) from Equation (15) and divide throughout by h_{++}^2 to give:

$$f_0' = \frac{\frac{f_{++}}{h_{++}^2} - \frac{f_{--}}{h_{--}^2} - \left[\frac{f_{+}}{h_{+}^2} - \frac{f_{-}}{h_{-}^2}\right]\left(\frac{x_{++}-x_{--}}{x_{+}-x_{-}}\right) - f_0\left\{\frac{1}{h_{++}^2} - \frac{1}{h_{--}^2} - \left[\frac{1}{h_{+}^2} - \frac{1}{h_{-}^2}\right]\left(\frac{x_{++}-x_{--}}{x_{+}-x_{-}}\right)\right\}}{\frac{1}{h_{++}} - \frac{1}{h_{--}} - \left[\frac{1}{h_{+}} - \frac{1}{h_{-}}\right]\left(\frac{x_{++}-x_{--}}{x_{+}-x_{-}}\right)} \tag{19}$$

5.3 Inputs

5.4 Outputs

6 NENZFr Response Surface

This code performs the calculation of a response surface and also handles the prediction of the nozzle exit flow. The response surface is constructed from a suitable set of NENZFr perturbation cases.

When completing a test campaign within a reflected-shock tunnel facility, the ability to calculate the nozzle flow properties within a few minutes of a shot is of great importance. The current implementation of NENZFr is not able to satisfy this requirement directly. Indirectly, the obvious solution and one that is frequently employed in similar situations is to use a set of NENZFr simulations to calculate a response surface describing the variation of the nozzle flow properties with the chosen inputs. Provided that it is sufficiently accurate, the response surface can be used to rapidly calculate the flow properties rather than running separate NENZFr simulations.

Based on the work of [6] and [7] two different response surfaces have been implemented within *nenzfr_RSA* and are detailed in the Section 6.2. In each case the response surfaces describes the variation of the nozzle exit flow properties over the normalised domain $(x_1, x_2) = (V_s/V_{s0}, p_e/p_{e0})$ where V_{s0} and p_{e0} are the incident shock speed and equilibrium nozzle supply pressure for the nominal condition on which the response surface is centered. Normalised coordinates were used to ensure that both variables were considered equally despite being different by four orders of magnitude ($V_s \sim 10^3$ while $p_e \sim 10^7$).

The choice to describe the variation of the nozzle exit flow properties in terms of only the incident shock speed and nozzle supply pressure was made based on the usual operation of shock tunnel facilities. When conducting an experimental campaign the tunnel may be operated at a number of different nominal conditions. For any given nominal condition, there exists shot-to-shot variability in the shock speed and nozzle supply pressure. This variation is typically small compared to that which exists over the range of available nominal conditions.

The intention is that the response surface will adequately account for the shot-to-shot variation of a given nominal condition. Consequently, separate response surfaces are required for each condition to be used during an experimental campaign. More often than not, creating the required response surfaces will be computationally cheaper than running individual NENZFr simulations for each shot.

6.1 Useage Examples

When creating a response surface *nenzfr_RSA* should be called from the same directory in which *nenzfr_perturbed* was ran. When using *nenzfr_RSA* to predict the nozzle exit flow, it only needs to be called from the same directory as the response surface data file. Below are some typical invocations.

- List all inputs and their default values:
  `nenzfr_RSA.py --help`
- Minimally build a 2nd-order response surface:
  `nenzfr_RSA.py --create-RSA`
- Build a 2nd-order response surface and calculate the residuals between the response surface and nenzfr cases on which it is based:
  `nenzfr_RSA.py --create-RSA --calculate-residuals`

- Use a response surface to predict freestream properties:
  ```
  nenzfr_RSA.py --Vs=2250.0 --pe=35.4e6
  ```
- Use a response surface to predict freestream properties and compare the results to a complete nenzfr simulation:
  ```
  nenzfr_RSA.py --Vs=2250.0 --pe=35.4e6 --calculate-residuals
  --exitStatsFile=../shot_10972/nozzle-exit.stats
  --estcjFile=../shot_10972/nozzle-estcj.dat
  ```
- Save the predicted freestream properties to a specified file:
  ```
  nenzfr_RSA.py --Vs=2250.0 --pe=35.4e6 --exitFile=10972-freestream.dat
  ```

Some additional notes:

- Don't use the radial-basis function response surface. It performs poorly. The 2nd-order response surface is a better method.
- When running *nenzfr_perturbed* use `--levels=3`. This produces the best (2nd-order) response surface which also performs quite well when extrapolating outside the design domain[7].
- Perturbing V_s by 5% and p_e by 15% when running *nenzfr_perturbed* in preparation for building a response surface should be appropriate for most nominal conditions. That is, the subsequent response surface should cover a sufficient region in the (V_s, p_e) space that extrapolation shouldn't be required in a typical test campaign.

6.2 Method

6.2.1 2nd-order Surface

For this option the variation of each property is modelled using a 2nd order polynominal equation of the form:

$$y^k = \beta_0^k + \beta_1^k x_1 + \beta_2^k x_2 + \beta_{11}^k x_1^2 + \beta_{22}^k x_2^2 + \beta_{12}^k x_{12} \tag{20}$$

for y^k in $\{T_s, h_s, \rho, v_x, v_y, p, \dots\}$ and normalised coordinates $(x_1, x_2) = (V_s/V_{s0}, p_e/p_{e0})$. Distinct values for $\{\beta_0, \beta_1, \beta_2, \beta_{11}, \beta_{22}, \beta_{12}\}^k$ exists for each y^k. Using a set of computed results, $\{(x_1, x_2, f^k)\}_i$, $i = 1 \dots n$, Equation (20) may be rewritten in matrix form as:

$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ y_n \end{bmatrix}^k = \begin{bmatrix} 1 & x_{11} & x_{21} & x_{11}^2 & x_{21}^2 & x_{11}x_{21} \\ 1 & x_{12} & x_{22} & x_{12}^2 & x_{22}^2 & x_{12}x_{22} \\ 1 & x_{13} & x_{23} & x_{13}^2 & x_{23}^2 & x_{13}x_{23} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{1n} & x_{2n} & x_{1n}^2 & x_{2n}^2 & x_{1n}x_{2n} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_{11} \\ \beta_{22} \\ \beta_{12} \end{bmatrix}^k \tag{21}$$

$$\mathbf{f}^k = X\boldsymbol{\beta}^k \tag{22}$$

This set of linear equations may easily be solved as follows[8]:

$$\begin{aligned} X^T\mathbf{f}^k &= X^T X \boldsymbol{\beta}^k \\ \therefore \boldsymbol{\beta}^k &= [X^T X]^{-1} X^T \mathbf{f}^k \end{aligned} \tag{23}$$

since $[X^T X]^{-1}[X^T X] = I$, the identity matrix.

[7]At least for the Mach 4, equilibrium gas test case.

[8]Multiplying by the transpose of X gives a square matrix of size $n \times n$ which can then be inverted

6.2.2 Radial-Basis Function Surface

For this method the variation of each property is modelled by interpolating over the entire set of computed results. That is,

$$y^k = \sum_i^n \beta_i^k \parallel \mathbf{x} - \mathbf{x_i} \parallel \tag{24}$$

for y^k in $\{T_s, h_s, \rho, v_x, v_y, p, \dots\}$, normalised coordinates $\mathbf{x} = (x_1, x_2) = (V_s/V_{s0}, p_e/p_{e0})$ and where $\parallel . \parallel$ represents the Euclidean distance and $\{(x_1, x_2, f^k)\}_i$, $i = 1 \dots n$ is a set of computed results.

Just as for the 2nd-order surface, the set of computed results can be used to find the distinct set of values $\{\beta_i\}^k$ for each y_k. In matrix form using the computed results Equation (24) becomes:

$$\begin{aligned}
\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_n \end{bmatrix}^k &= \begin{bmatrix} 0 & \parallel \mathbf{x}_1 - \mathbf{x}_2 \parallel & \parallel \mathbf{x}_1 - \mathbf{x}_3 \parallel & \dots & \parallel \mathbf{x}_1 - \mathbf{x}_n \parallel \\ \parallel \mathbf{x}_2 - \mathbf{x}_1 \parallel & 0 & \parallel \mathbf{x}_2 - \mathbf{x}_3 \parallel & \dots & \parallel \mathbf{x}_2 - \mathbf{x}_n \parallel \\ \parallel \mathbf{x}_3 - \mathbf{x}_1 \parallel & \parallel \mathbf{x}_3 - \mathbf{x}_2 \parallel & 0 & \dots & \parallel \mathbf{x}_3 - \mathbf{x}_n \parallel \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \parallel \mathbf{x}_n - \mathbf{x}_1 \parallel & \parallel \mathbf{x}_n - \mathbf{x}_2 \parallel & \parallel \mathbf{x}_n - \mathbf{x}_3 \parallel & \dots & 0 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \\ \vdots \\ \beta_n \end{bmatrix}^k && (25) \\
\mathbf{f}^k &= X\boldsymbol{\beta}^k \\
\therefore \boldsymbol{\beta}^k &= X^{-1}\mathbf{f}^k && (26)
\end{aligned}$$

since X is a square matrix of dimension $n \times n$.

6.3 Inputs

6.4 Output Files

7 NENZFr Quasi-Transient

7.1 Method

7.2 Inputs

7.3 Outputs

References

[1] M. K. McIntosh. Computer program for the numerical calculation of frozen and equilibrium condtions in shock tunnels. Technical report, Department of Physics, Australian National University, Canberra, A.C.T., September 1968.

[2] J. A. Lordi, R. E. Mates, and J. R. Moselle. Computer program for the numerical solution of nonequilibrium expansions of reacting gas mixtures. Contractor Report 472, NASA-CR-472, 1966.

[3] Sandford Gordon and Bonnie J. McBride. Computer program for calculation of complex chemical equilibrium compositions and applications 1. analysis. Reference Publication 1311, NASA-RP-1311, October 1994.

[4] D.J. Mee. Uncertainty analysis of condtions in the test section of the t4 shock tunnel. Departmental Report 1993/4, Division of Mechanical Engineering, Unversity of Queensland, Brisbane, Australia, 1993.

[5] Rainer Kirchhartz. *Upstream Wall Layer Effects on Drag Reduction with Boundary Layer Combustion.* Phd thesis, School of Mechanical and Mining Engineering, Centre for Hypersonics, The University of Queensland, Brisbane, Australia, 2009.

[6] Timothy W. Simpson, Dennis K. J. Lin, and Wei Chen. Sampling strategies for computer experiments: Design and analysis. *International Journal of Reliability and Applications*, 2(3):209–240, 2001.

[7] Anthony A. Giunta, Steven F. Wojtkiewicz, and Michael S. Eldred. Overview of modern design of experiments methods for computational simulations. In *41st Aerospace Sciences Meeting and Exhibit.* AIAA-2003-0649, January 2003.