ARL-SR-0515 ● Feb 2026

IBHVG User Manual

by Andrew McBain

NOTICES

Disclaimers

ARL-SR-0515● FEB 2026

IBHVG User Manual

Andrew McBain
DEVCOM Army Research Laboratory

REPORT DOCUMENTATION PAGE

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED START DATE	3. DATES COVERED END DATE
February 2026	Special Report	1 July 2024	30 September 2025

4. TITLE AND SUBTITLE
IBHVG User Manual

5a. CONTRACT NUMBER	5b. GRANT NUMBER	5c. PROGRAM ELEMENT NUMBER

5d. PROJECT NUMBER	5e. TASK NUMBER	5f. WORK UNIT NUMBER

6. AUTHOR(S)
Andrew McBain

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
DEVCOM Army Research Laboratory ATTN: FCDD-RLA-WC Aberdeen Proving Ground, MD 21005	ARL-SR-0515

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.

13. SUPPLEMENTARY NOTES
ORCID: Andrew McBain, 0000-0002-9578-1457

14. ABSTRACT
Interior Ballistics of High-Velocity Guns (IBHVG) is a lumped parameter interior ballistic code developed by ARL in support of its research into propellant modeling and simulation. This manual provides a brief introduction into interior ballistics as well as a detailed explanation of the different keywords and options in the code IBHVG. This new release of IBHVG (version 5.1) fixes several bugs, some going back decades, and presents a more modern operation for the computers of today. The manual was also updated with some new sections, including a quick start guide, a short lesson on the most important variables for the modeler to consider, and how those values might be determined.

15. SUBJECT TERMS
Weapons Sciences, interior ballistics, propellant, modeling and simulation, software documentation

16. SECURITY CLASSIFICATION OF: a. REPORT	b. ABSTRACT	C. THIS PAGE	17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UU	69

19a. NAME OF RESPONSIBLE PERSON	19b. PHONE NUMBER *(Include area code)*
Andrew McBain	(520) 691-6438

STANDARD FORM 298 (REV. 5/2020)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

List of Tables

1. Introduction

Welcome and thank you for using Interior Ballistics of High-Velocity Guns (IBHVG) (and reading the manual). For new users, everything you need to get started should be contained in this manual. For experienced users, please pay special attention when running legacy decks with the current version of IBHVG. Many changes have occurred; while backward compatibility has been a desired goal, sometimes it was not possible, and in some cases that compatibility was purposely broken (to fix a bug or to make a feature more useful). A list of some of these changes can be found in Section 3, Changes. For any user, please do not hesitate to reach out to the email support mailbox,* whether it is for advice on how to 1) model a particular system, 2) debug an input deck, 3) tune a model, or 4) report a bug.

IBHVG version 5.1 is based on IBHVG v2.507 with several additions made as well as most of the features from IBHVG v3 backported. This document serves as a user manual for IBHVG and the basics of interior ballistics (IB). Some components of this documentation were built upon BRL-TR-2829[1] and the document by Anderson.[2] The basics of user input and operation are covered as well as details on where to gather information needed to construct an interior ballistic model. Some annotated examples are included with the code as a guide to the user as well as a basic input deck included under the Quick Start section (see Section 2.2). It is highly recommended that this document is not the sole learning material by which to learn IB and that a new user in this area seek out an experienced mentor. If there is any hesitancy in how to proceed, please consult an individual who is experienced in IB before using this program for any study or design work. Feel free to reach out to the Weapons Propulsion Science Branch at the U.S. Army Combat Capabilities Development Command (DEVCOM) Army Research Laboratory (ARL) (either by contacting the branch chief† or by the email address*) for bugs, advice, or guidance or to discuss potential feature requests. For government researchers requiring the most up-to-date version of this manual, please consult the 'docs' folder in the IBHVG Git repository. For those without access, updated manuals with corrections will be released semi-regularly and/or timed with new features.

Some new material or features to IBHVG have been deemed critical technology and are documented in a separate CUI-level Appendix ARL-SR-0516.[3] The Appendix will be periodically updated and released alongside this public release user manual.

---

* usarmy.apg.devcom-arl.list.ibhvg@army.mil

† michael.a.minnicino.civ@army.mil

1.1 Compiling IBHVG v5.1

Compiling of IBHVG can be done via the command

'make FC=ifort'

from the ./src/ directory of the repository. This will compile IBHVG using the Intel Fortran Compiler.

Please refer to the Makefile in the IBHVG repository on the HPC GitLab for details on compiling IBHVG, including options for compiling with GCC.

Some of the flags included were for ease of debugging and are not necessary for the operation of IBHVG. Several warnings will be produced, and most of these are about formatted print spacing.

If an 'objs' folder is generated during compiling, it is needed to operate IBHVG and must be located in the same directory as the executable.

1.2 Assumptions

IBHVG is a "lumped parameter" (i.e., thermodynamic code) for describing interior ballistic systems. As such, it has little spatial knowledge of the gun system and relies instead on an assumed pressure gradient to describe the state of the gas throughout the combustion chamber.

IBHVG also ignites all the propellant at the start of the simulation (unless manually specified). Thus, IBHVG will never generate a curve that overlays experimental data without some shift in time. Matching peak pressures in time can often give a unique view into how the simulation compares in the ignition phase (at times before peak pressure) and as the propellant burns out and the gas primarily expands (at times after peak pressure).

1.3 Limitations

IBHVG cannot predict negative differential pressures (NDPs) in a gun system. NDPs are the result of a localized increase in gas production[4,5] and can occur for several reasons:

- Propellant fracturing and thereby having more surface area than the surrounding areas
- Ignition delay causing some localized area to be ignited with a nearby area not producing gas
- Large amounts of ullage (empty space) with burning propellant not uniformly distributed

None of these scenarios can be modeled in IBHVG because a core assumption is a predefined distribution of gas through the chamber (i.e., the Lagrange gradient, its special cases, or the Pidduck–Kent gradient).

Therefore, it is highly recommended to use GRAD=2 or =3 in INFO because it makes simulations of systems with large charge-to-projectile mass ratios and chambers without straight walls more accurate.

IBHVG does not provide an effective way to design igniters. This is due to the primer being completely burned and the product gases uniformly distributed throughout the chamber at the start time of the simulation. Even for parametric trade studies to estimate a required igniter mass, IBHVG does a poor job. It is good practice to only specify a fraction of the primer in the actual primer deck and include the rest as a propellant to burn over time.

WARNING! A system that cannot be modeled successfully with IBHVG should be carefully studied before operation because dangerous conditions are much more likely outside of the assumptions that underpin IBHVG. Higher fidelity IB modeling with computational fluid dynamics (CFD) offers a means of investigating these two limitations and requires a corresponding increase in system details as input and analysis of the results.

2. Quick Start

2.1 Basic Operation

Data is inputted into IBHVG by using a plain text file. Some of the nomenclatures (and features!) are remnants of IBHVG's past use on older computers that used punch cards for input.

Unlike in IBHVG v3, the standard input and output are not used during operation. Instead, command-line arguments are used, and the nomenclature should be as outlined below.

2.1.1 Windows

```
Run ibhvg.exe input_textfile.in output_textfile.out
```

2.1.2 Linux/UNIX

```
./ibhvg.exe input_textfile.in output_textfile.out
```

The file extensions on the input and output files do not matter because IBHVG will read and/or output the formatted text.

The text in the input files is separated into distinct sections and referred to as decks. The beginning of a deck is indicated with a $ in the first column of the first line and followed by up to four alphanumeric characters to indicate which section of input is described in the following lines. An additional parenthesis can follow containing a number to indicate the specific deck [e.g., $PROP(3)]. This is especially useful when using the SAVE command.

Inside of a deck, data is entered by entering the name of a variable followed by an equals sign and the value to be entered. Text stings require single quotes to enclose the string. Arrays are assumed to start at their first index and ascend in index for each additional entry. In the case of arrays, an index can be specified in parenthesis after the keyword. The following is an example of the different ways to structure the keywords in an input file.

```
DELT = 0.001
TITL = 'TEXT INPUT'
POPT = 1,2,1
POPT(4) = 0,1,1
PRL(1,1) = 1000
```

In the previous example, the variable DELT is filled on row 1, TITL on row 2, the first three indexes of POPT on row 3, the fourth, fifth, and sixth indexes of POPT on row 4, and the index (1,1) of PRL on row 5.

Multiple variables can be put on a single line as long as the total data entry for the line does not exceed 100 characters. Adding '$' will indicate the end of useable data entry, and anything after this character will be excluded from being read. This is useful to place notes or comments about the input deck in-line with the value in question. As the '$' stops the read on that line before the 100-character limit, the comment can exceed it on that line because it is not useful information for the IBHVG simulation. Data entry can "wrap" around lines because internally the code reads all lines in a deck and then concatenates them into a single sequence to parse.

Although it is not required, it is advisable to put different variables for input on their own line for legibility and allow for informative comments, e.g., noting from where the value came (technical document, communication, date, etc.) (Figure 1).

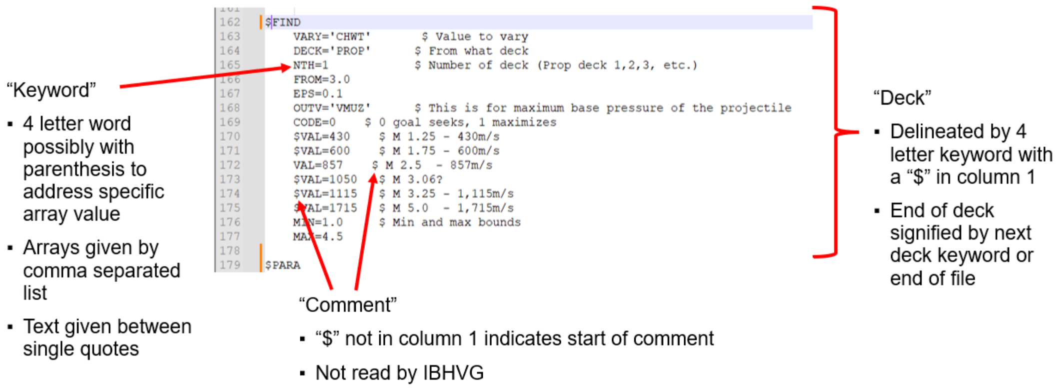

Figure 1. An annotated deck with commented out components, different text alignments, and what they indicate.

2.2 The Minimum

The minimum information required to start a simulation is as follows:

```
$INFO
    $ No information is actually needed in the INFO deck
    $ However it must be present to initialize the simulation
    $ There are useful variables in here for controlling the output

$GUN
    $ Defines physical dimensions of the combustion chamber
    CHAM = ####     $ [m^3] Chamber Volume in cubic meters
    LAND = ####     $ [m] Bore diameter in meters
    TRAV = ####     $ [m] The distance the projectile travels
    $ TRAV is one primary control on the length of the simulation

$PROJ
    $ Defines the projectile properties for the simulation
    PRWT = ####     $ [kg] Projectile mass in kilograms

$PRIM
    $ Provides initial energy to the simulation
    CHWT = ####     $ [kg] Primer mass in kilograms

    FORC = ####     $ [J/kg] Primer combustion products impetus
    TEMP = ####     $ [K] Primer combustion flame temperature
    GAMA = ####     $ [no units]
  $ Primer combustion products ratio of specific heats (cp/cv)
    COV  = ####     $ [m^3/kg]
  $ Primer combustion products covolume (non-ideal gas corrective term)

$PROP
    CHWT = ####     $ [kg] Propellant mass
    GRAN = 'str'    $ [text] Propellant shape description (form function)
```

```
    PD   = ####    $ [m] These will change
    WEB  = ####    $ [m] depending on the shape
    LEN  = ####    $ [m] of the propellant chosen

    BETA = ####    $ [m/s/MPa^n]
  $ Propellant burning rate coefficient as needed by Vieille's Law
    ALPH = ####    $ [no units]
  $ Propellant burning rate exponent as needed by Vieille's Law

    RHO  = ####    $ [kg/m^3] Propellant solid phase density
    FORC = ####    $ [J/kg] Propellant combustion products impetus
    TEMP = ####    $ [K] Propellant combustion flame temperature
    GAMA = ####    $ [-]
  $ Propellant combustion products ratio of specific heats (cp/cv)
    COV  = ####    $ [m^3/kg]
  $ Propellant combustion products covolume (non-ideal gas corrective term)

$END
```

Note that if the line wraps, it is an artifact of the conversion from markdown. IBHVG will not recognize a comment that wraps to a new line and could potentially throw an error or crash (or work, it could do that too).

For more information on a given deck, follow the links: INFO, GUN, PROJ, PRIM, PROP, and END.

The above example assumes no heat loss from the combustion chamber, which can be added simply through the inclusion of 'HL=1' in HEAT and which can have a moderate effect on the simulation. Fine tuning this parameter beyond setting HL=1 can be difficult.

The above example also assumes no bore resistance (including any air resistance), which can be added through the inclusion of RESI and can have a moderate-to-large effect on the simulation.

Although this is just the minimum information needed to run a simulation, often including more information about the system can significantly increase the fidelity of the simulation. Likewise, a more detailed representation of some of the information presented (i.e., a more complex representation of the propellant burning rate instead of a single global relationship) can also improve results.

The output of IBHVG can be controlled by the INFO deck, which includes options to print only specific variables via TDIS decks or generating a .csv output for computer parsing (POPT(10)=1). *It is highly recommended that the print echo (POPT(1)=1) be checked to verify the information entered is the information that*

IBHVG is using in the simulation, and especially in cases where the code is manipulating values itself (i.e., PARA, FIND).

For new users to IBHVG and IB, it is recommended to read Section 5, Interior Ballistics, as well before creating an input deck because it offers some insight into how to model gun systems.

2.3 Common Errors

Using a text editor to develop an IBHVG deck is highly encouraged. Microsoft Word is not a text editor and can add characters and unknown formatting to the file that can cause IBHVG to not run properly. Notepad++ is available through the software center at ARL and will work, and it includes many advanced features, although vi, emacs, or even regular Notepad are equally functional.

When using a text editor, please avoid the use of tabs (it is likely you can set your text editor of choice to replace any inputted tabs with a multiple of spaces automatically).

Due to how IBHVG v5.1 reads an input file, it will no longer crash on reading a particular value. Instead, it attempts to emulate this behavior, but it is no longer guaranteed that an input error takes place immediately after where the code stopped. An error in the input will cause IBHVG to attempt to guide the user to the nature and cause of the error via a message in the output file. This is because IBHVG concatenates the entire deck sans comments to parse the contents as opposed to the line-by-line operation of IBHVG v3 and prior versions. Furthermore, IBHVG now reads the input file three times (one pass for figuring out the number of decks, once more to identify array sizes, and a final time to fill the newly formed arrays), and each read can theoretically produce its own new issues at different locations.

Remember! The most common and pernicious error is user error by inputting the incorrect values to the simulation. This may not produce an error message and may even generate an answer (albeit an incorrect one). Double-check the input, preferably via the echoed input of your simulations for units, diameters (not radii), and typos to avoid this issue.

3. Changes from Past Versions of IBHVG

3.1 Input Decks

The IBHVG input file parser was completely rewritten for IBHVG v5. This was done to facilitate unlimited/user-defined array sizes. A considerable effort was made to ensure backward compatibility with all IBHVG decks. However, it is possible that read failures will occur, which are often related to poorly organized, written, or nonstandard input decks. Some features were rewritten and therefore minor input changes might be needed, or different results may occur (e.g., FIND decks, in particular the EPS variable). *Please refer to the IBHVG Input Decks sections and carefully double check nothing has changed for decks that are reused from legacy IBHVG v3 and are essential*. In addition, IBHVG v5.1 stops data input when it reaches the end of the input file, thereby deprecating the END command and requiring that separate input decks be contained in separate files.

3.2 Post-Simulation Analysis

IBHVG now stores all time steps of the simulation in memory until completion, whereupon it is printed to the output file. Previous versions of IBHVG wrote out each step of the simulation to the file and only kept the most recent version in memory. Now, analysis of the various trajectory variables can be completed and outputted at the time of simulation through routines employed internal to the code. The use of the new print options or the Python interface also allow simplified post-simulation analysis for features not yet included.

3.3 Tapered Bore

Added to the unreleased IBHVG v4 was a feature to simulate tapered bore guns. This was an extension of the chambrage pressure gradient introduced in BRL-TR-3097.[6] To use the tapered bore pressure gradient option, the GRAD variable in the INFO deck is set to 4, and the CPTS, DIST, DIAM in the GUN deck should define the gun from breech to muzzle, instead of the start of projectile travel. In this case, the projectile start of travel is defined by integrating the volume generated by the DIST and DIAM parameters until the inputted CHAM volume is reached. The ability to run a simulation to maintain a certain projectile acceleration is available in the POPT array of the INFO deck. This is achieved by reducing the bore diameter from the value at the seating depth to the final diameter in the DIAM array. Changes have been made to these subroutines from IBHVG v4 (unreleased) for IBHVG v5 to reduce the memory needed by calculating the required integrals as needed at the exact projectile location instead of creating a lookup table for interpolation.

3.4 Units and Files

IBHVG v5 was initially based on IBHG v2.507 and imperial unit options were re-added to the code. In IBHVG v5, all calculations are done in base metric units, i.e., N-m-sec and K. All other units are derived from the base metric units. Imperial input units are converted to metric after read-in, and any imperial unit print out is done for output reading only. This is most notable in the TRAJ deck variables where some default units have changed (e.g., MPa are now Pa).

3.5 Form Functions

A new form function was added to IBHVG to provide a different method for modeling deterred propellants. The QLOD form function is interoperable with parameters from the commercial code QuickLOAD. This is not an exact mapping because IBHVG requires the propellant's flame temperature, covolume, and so on instead of using the heat of explosion. In addition, the GEN form function has the new option to specify a number of grains to be used (instead of one, single, large grain) in conjunction with the surface area by depth table. In addition, the initialization of practically every form function was redone to support better operation with PARA and FIND (which historically could be buggy) as well as provide easier use of ratios and warnings when IBHVG makes an assumption.

3.6 Multiple RESI Decks

IBHVG can now handle multiple RESI decks. These are summed up at runtime and the output is printed in the input echo. It is not necessary for the RESI decks to align to each other because IBHVG will interpolate and add new points as needed. This can be useful when different effects are trying to be modeled via the bore resistance. By using multiple RESI decks, it can be easier to vary one effect while keeping others constant.

3.7 PROP Deck Linking

The PROP deck now includes a variable 'NTH,' which allows multiple PROP decks to share a value while maintaining other unique values. For example, one propellant whose charge weight is split among several PROP with differing ignition times but maintaining a consistent WEB across them such that a PMAX deck can be used to optimize it to hit a specific breech pressure.

3.8 PLOT

IBHVG now supports making plots of the IB models and simulations it can run. It does this by interfacing with the software gnuplot, which is freely available and does not require installation. Rudimentary plotting is available by using the variables listed below in the PLOT deck description, or a plot file can be made externally and pointed to gnuplot to use instead.*

3.9 DEBUG

A new deck, DBUG, allows the operator to better debug the inner workings of IBHVG by turning on and off various levels of detailed reporting on the functionality of individual subroutines. This is possibly more helpful to the experienced IBHVG user, or even the IBHVG programmer, than the novice user. Debugging the input reader can also be enabled by using a third command line argument, for example,

```
./ibhvg.exe test.in test.out third
```

where the third argument does not need to be anything more than a single character. This was added to debug errors before the keywords were read-in and parsed.

3.10 TDIS Output Precision

You can now specify the output precision of TDIS variables using the PREC variable under the TDIS deck. For example, it allows pressure to be printed in pascals and output to be generated to the single pascal (as opposed to the kilopascal, per se).

3.11 FIND Rewrite

The FIND algorithm was replaced with the Nedler–Mead/Downhill Simplex method with a heavy amount of modification from Press et al.[7] Although it is less efficient than the previous method, it accomplishes multidimension optimization seemingly more robustly. The FIND deck description has more details on the updated use and operation of this feature.

3.12 TRAJ Parameters

New TRAJ parameters were included to support the various new decks and calculations that an interior ballistician may need to perform.

* gnuplot, http://www.gnuplot.info/

3.13 Array Sizes

There are no fixed array sizes. All arrays require a user input to specify array size, and the required space is allocated at runtime. This works for implicit array sizes as well, similar to the number of propellants or TDIS variables to print.

3.14 Gun Recoil with Chambrage

Gun recoil that is simulated when using either the chambrage gradient or the tapered bore gradient now accounts for the differing pressure on the various parts of the chamber (i.e., the forcing cone reducing the recoil felt as per a free body diagram) (Special thanks to Brett Sorenson from ARL's Lethal Mechanisms Branch for identifying this bug.) This does not affect recoil during blowdown at this time because that is calculated via empirical fit, and a method for accounting for the difference has not been developed at this time.

3.15 PMAX On/Off (For Use With SAVE)

A new parameter was added to the PMAX deck to turn off a PMAX run from inside its own deck. This is useful in combination with the SAVE function to find a propellant property to hit a PMAX and then perform iterations on that value without continuing the PMAX calculation. The use-case presented optimizes a web with a burn rate for a "hot" temperature and then calculates the performance loss at "ambient" and "cold."

3.16 Removal of Some Legacy or Half-Attempted Features

There are errors people might encounter from legacy decks, often related to the removal of the underlying Common blocks in the source code:

- Preallocated arrays in IBHVG v1/2/3 allowed data input beyond what was specified as long as it was within the preallocated bounds. For example, IBHVG v5 will crash if NGAG = 5 and GLOC(6) is specified, but IBHVG v3 will happily run the simulation. Likewise, leaving the array bound undeclared (i.e., not defining NGAG) but still declaring GLOC(6) will cause IBHVG v5 to crash, whereas IBHVG v3 will happily run.
- Similarly, it is possible for IBHVG v5 to develop a memory leak. Although all efforts have been made to detect and fix memory leaks if during particularly long simulations you see that IBHVG is using more and more compute resources (i.e., RAM), please bring it to the IBHVG team's attention so that it can be fixed. These issues were impossible with

IBHVG v3 because the initial memory was defined from the outset. By switching to user-defined array sizes, this type of issue is now possible.

4. Python Interface for IBHVG

A python interface for IBHVG is included in the GitLab repository to facilitate IBHVG's use in more complex and automated workflows. It currently uses the same key words for each class as attributes for each deck as part of its input class. If there are multiple key words, then the attribute used is identified in italics. Key words that were unsuitable used the unique key word identified in parentheses. All key words are in lowercase (as seen in the example below).

There is no need to fill out the key words related to array sizes because they are handled automatically at runtime. In addition, each deck includes a 'note' keyword to allow comments to be placed into the deck. The 'note' can be either a string or a list of strings.

An example of its use is shown below:

```
from IBHVG_Classes import IBHVG
import os

sim1 = IBHVG()

gibm = r'.\IBHVG\ibhvg-x86_64-w64-mingw32.exe'
run_dir = os.path.dirname(gibm)

sim1.path = gibm
sim1.file_in = os.path.join(run_dir,'IBpy.in')
sim1.file_out = os.path.join(run_dir,'IBpy.out')

sim1.input.info = sim1.decks.info()
sim1.input.info.popt[0] = 1  # watch out as these get reindex to 0 bein
g first...
sim1.input.info.popt[1] = 1
sim1.input.info.popt[2] = 1

sim1.input.comm = sim1.decks.comm()
sim1.input.comm.text = 'This example is similar to the Quick Start'

sim1.input.gun = sim1.decks.gun()
sim1.input.gun.cham = ####
sim1.input.gun.land = ####
sim1.input.gun.trav = ####

sim1.input.proj = sim1.decks.proj()
sim1.input.proj.prwt = ####
```

```
sim1.input.prim = sim1.decks.prim()
sim1.input.prim.chwt = ####
sim1.input.prim.forc = ####
sim1.input.prim.temp = ####
sim1.input.prim.gama = ####
sim1.input.prim.cov = ####

sim1.input.prop = sim1.decks.prop()
sim1.input.prop.name = 'propellant'
sim1.input.prop.chwt = ####
sim1.input.prop.gran = 'form function'
sim1.input.prop.pd = ####
sim1.input.prop.web = ####
sim1.input.prop.len = ####
sim1.input.prop.rho = ####
sim1.input.prop.forc = ####
sim1.input.prop.temp = ####
sim1.input.prop.gama = ####
sim1.input.prop.cov = ####
sim1.input.prop.prl = [[####, ####]]  # multidimensional as [[,,],[,,],
[,,]] etc.
sim1.input.prop.cfl = [[####,####]]
sim1.input.prop.exl = [[####,####]]

# sim1.input.END = sim1.decks.end()  # End is appended at end if one is
not included

# sim1.order = ['info','comm','gun','proj','prim','prop','END']  # not
yet supported

sim1.run()
```

Planned additions are to be able to read existing IBHVG files to populate the input structure. As part of that, it may require some refinement to support multiple decks (like PROP) and making sure they are printed in a desired order (e.g., for SAVE). It does not reread the echo (or a machine readable version thereof), and so it does not know if input values were changed (i.e., from a PMAX deck). It also does not support any unit checking or conversions at this moment.

Because of the utility of Python, there are several IBHVG features that may not be needed (see PLOT, FIND, SAVE, etc.). The simplest operation might be to use IBHVG for just the IB calculation and leave all other components to different Python packages.

5. Interior Ballistics

5.1 Introduction

IB is chiefly concerned with the *rate* of energy conversion. That is, the speed at which the solid propellant is turned into gas to push the projectile down the bore. If the rate of energy release is too slow, then the projectile will not have enough kinetic energy to reach or affect the target. If the rate is too fast, then the pressure built up in the chamber can exceed the strength of the gun and result in a dangerous condition (i.e., the gun blows up!).

IB is a balancing act of providing just the right amount of energy to build up pressure before the projectile accelerates without blowing up the gun and then trying to maintain that pressure. With traditional propellant shapes and charge designs, the maximum pressure is the point at which the projectile accelerating opens up too much volume for the charge to produce enough gas to fill at the same pressure (and thereby the roll off and then decrease in pressure).

Often, IB is talked about as the "broadening the curve." This refers to the pressure curve generated over the time of the gun firing event. Integrating the area under the pressure curve is proportional to the work done on the projectile. Therefore, a curve that reaches the same maximum pressure but is broader will have a greater muzzle velocity or energy.

Although the maximum pressure is important, there are often several different criteria that are important to calculate:

- Muzzle velocity (e.g., artillery zoning)
- Maximum acceleration (e.g., fuze requirements)
- Muzzle pressure (e.g., blast overpressure on operators)
- Recoil impulse (e.g., energy transferred to weapon platform)

A simulation in IBHVG can provide values for all of these criteria and more.

5.2 Energy Conversion

The essential part of an IB simulation is the conversion of the solid propellant into the gas that propels the charge. It has been shown that solid propellant can pyrolyze and then undergo a complex series of chemical reactions in its conversion to a gas[8]; however, IBHVG does not simulate that. Instead, the solid propellant is "burned" and replaced with an equal mass of predefined gaseous products. These are described by the thermochemical properties that were entered into the PROP deck:

- Impetus
- Flame temperature
- Covolume
- Ratio of specific heats (gamma)

Some of these are related (Impetus = Ideal Gas Constant * Flame Temperature/Product Species Molecular Weight) and (Chemical Energy = Imeptus/(Gamma - 1)) and may be used in differing forms in different codes.

Nevertheless, the gas generation rate is defined as $dm_g/dt = rho * SA * b_r$, where rho is the solid density, SA is the current burning surface area of the propellant, and b_r is the burning rate (often provided according to Vieille's law, βP^{α}, where the Pressure is the Mean Chamber Pressure)

From this, we can tell that the most important factors for understanding the energy release is the current surface area of the propellant and the burning rate (as a function of pressure). Burning rate is something over which the charge designer has little control; it is a property of the propellant to be used. Different propellants exhibit different burning rates, so although some control is available, it is not nearly fine enough on its own.

As an aside, when it is said that "burn rate is a material property," that is somewhat idealized. The burn rate used in IBHVG is often from a closed bomb test, which can be highly system-dependent in some cases (i.e., when a tactical geometry is used as opposed to a large single perforated [perf] or large discs of propellant). In addition, there may be erosive effects in tactical propellant geometry that may not be measured if a strand burner is used instead of the closed bomb. It is debatable how much of this is attributed to the material property versus used to cover up for other shortcomings in IB simulations.

Thus, what remains to control the energy release rate is the surface area of the propellant. This is done through careful control over its initial shape. By adding holes into the propellant (perforations), the surface area can be made to increase as a function of burn depth (i.e., the hole expands) (Figures 2 and 3). IBHVG assumes that the propellant regresses (i.e., burns) normal to the solid surface into the bulk of the grain following Piobert's law. The volume of the grain burned is taken as surface area at the start of the integration time multiplied by the burn depth (as opposed to XKTC or NGEN, which recalculate the volume of the shape to find the difference). The latter approach is more accurate and will likely be adopted by IBHVG in the future.

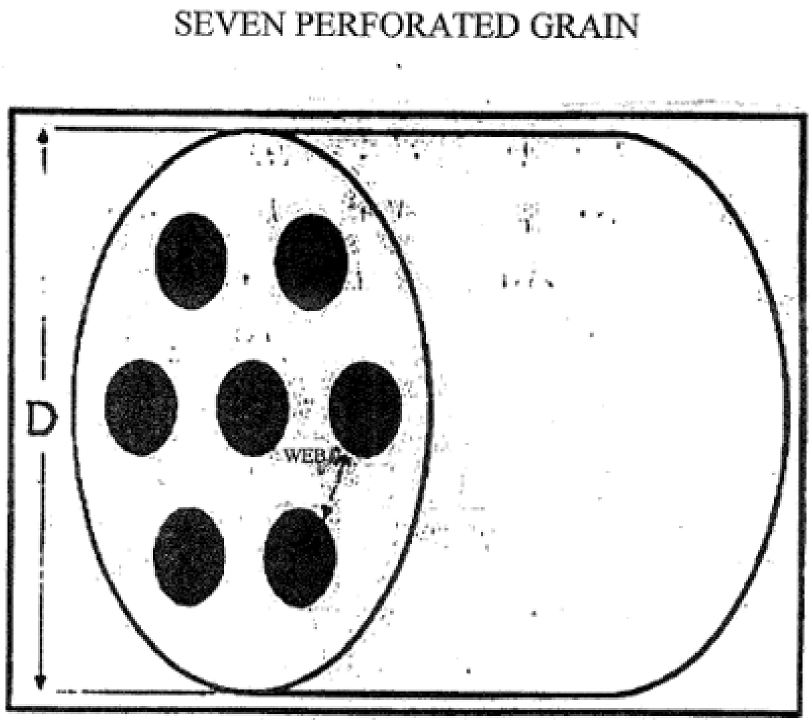

Figure 2. The shape of a "seven perf" (or 7P or 7PF) grain.[9]

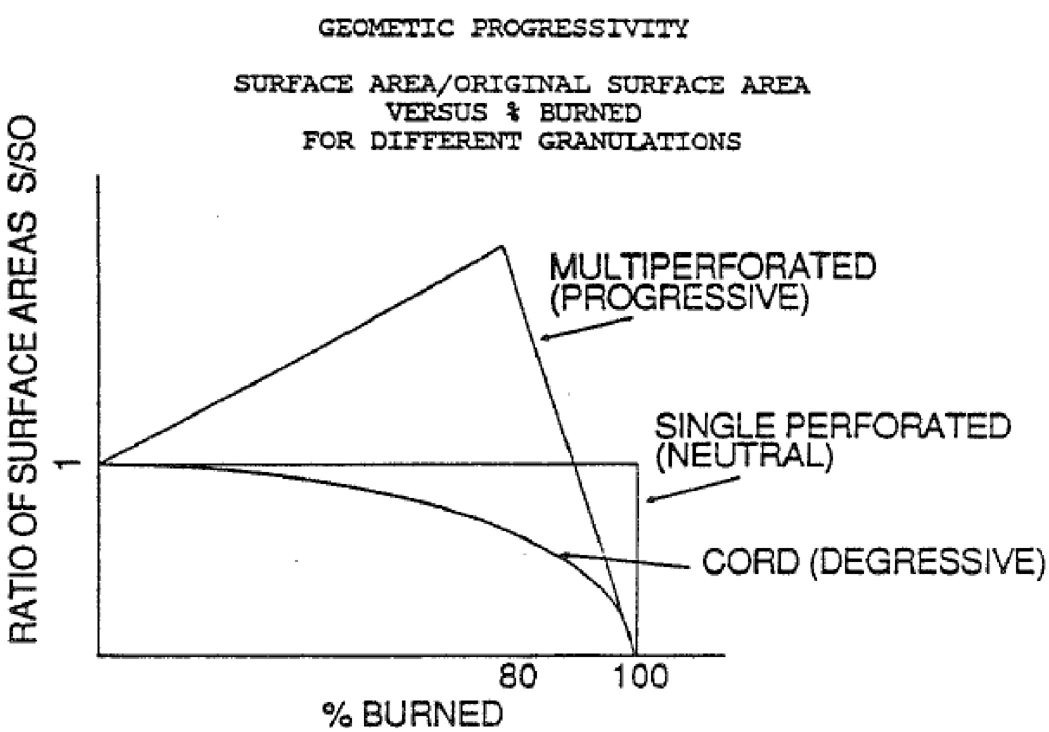

Figure 3. The different form function progressivities.[9]

By selecting an initial perforation diameter and web (spacing between the perforations), and in conjunction with its burn rate, the propellant can be tailored to optimize the charge for its specific needs.

In small caliber systems, fine control over the surface area is less feasible, so instead chemical diluents (or more commonly deterrents) are diffused into the propellant. This results in a geometrically regressive shape but an increasing amount of energy released (and often increase in burning rate) as a function of depth. This is markedly more difficult to quantify and has additional aging concerns as the diluent migrates through the solid propellant over time. IBHVG supports modeling deterred propellants in addition to geometrical controlled ones via propellant layers.

5.3 The Important Things to Get Right

Thus, from the information above, and that provided in the Quick Start, a sense of the most important components of an IB simulation can be found.

- Characterizing the surface area of a propellant as a function of depth (its form function)
- Characterizing the burn rate
- Gun chamber dimensions/volume
- Projectile mass

6. IBHVG Input Decks

6.1 COMM Deck Reference

A comment deck follows this deck name, and all the lines are read without further interpretation until the next deck name is encountered ($ in column one).

6.2 DBUG Deck Reference

This deck allows access to the internal debugger of IBHVG, which outputs to the file 'IB DUMP.txt'. All DBUG parameters have a default value of 0. To enable DBUG output, add the parameter name and an integer > 0. Some routines support additional output levels over the base value of 1 with greater integer values (2,3,5, etc.). The debug statements found within are not exhaustive but may prove helpful in narrowing down bugs or calculation issues (Table 1). In addition, adding anything as a third input argument when calling IBHVG activates the debug mode for the input file IO and parsing; this cannot be activated by the input file because it has not been read yet. Enabling multiple commonly called subroutines (i.e., anything called from DERIV1) can lead to very large output file sizes if the code is hanging or stuck in a loop. In this case, it is best to manually kill the program because the error can likely already be spotted in the dump file.

Table 1. DBUG keywords.

Parameter name	Description
RESCOMBINE	Called from MODRHESAPE (in the input portion of the code). Combines multiple RESI decks.
UNITS	Called from INPUTDATA. Converts some units on file input. Also calls M2E.
M2E	Called by UNITS and PRINTER. Converts from metric to English units, and back. Every calculation in IBHVG is done in metric base units and the input and output adjusted when it is read-in or written out.
DRIVER	Called from the main program. Controls multiple IB runs via calling MXBRPR (PMAX run), AMOEBA (FIND runs), or INTBAL (plain single run) as well as PARA searches.
MXBRPR, PMAX	Called from DRIVER. Controls IB runs for PMAX searches.
AMOEBA, AMOTRY, FNEVAL2, FSTORE, FIND	Called from DRIVER. Controls multiple subroutines that all work together to complete FIND IB simulations. AMOEBA controls the overall downhill simplex method, AMOTRY is called from there, FNEVAL2 evaluates the closeness of each amoeba vertex, FSTORE stores each run.
INTBAL	Called from DRIVER, MXBRPR, AMOEBA. Controls start, stop, printing, everything for a single run/IB simulation (e.g., FIND calls INTBAL multiple times). Some minor cleanup occurs at the end of each IB run and that can also be found here.
PRPDLAY	Called from INTBAL. Establishes new PROP decks for negative IGNC values.
IBPREP	Called from INTBAL. Verifies input and initializes flags and values for a single IB run.
STDPRP	Called from IBPREP. Calls PRPSETUP and PRPSETUP2. Called from IBPREP, mainly. Establishes a propellant. A value of 2 or higher will print a text file with a complete user and/or machine-readable PROP deck.
PRPSETUP	Called from STDPRP. Establishes a PROP propellant. Also turns on output for PRPSETUP2, which establishes a PR2 propellant.
BNDCON	Called from STDPRP2. PR2 boundary conditions and calculation.
CBHLOSS	Called from IBPREP. Calculates heat loss for closed bomb simulations.
INITPR	Called from IBPREP. Initializes flags and values for a standard IB run.

Table 1. DBUG keywords (continued).

Parameter name	Description
INITCP	Called from IBPREP. Initializes flags and values for a constant pressure run. Routine requires double-precision flag when compiling; otherwise, it will get stuck in an infinite loop.
SHOTGO	Called from IBPREP. Determines if the projectile is moving at the beginning of the simulation.
PICKDT	Called from INTBAL. Calculates maximum integration step size.
KUTMER	Called from INTBAL. Main integration routine. 5-step Runge-Kutta so all derivatives are calculated 5 times per main time step.
KMTRM1	Called from KUTMER. Calculates values representing transition conditions for KUTMER.
DERIV1	Called from KUTMER. Calculates derivatives for KUTMER. Prints 5×KUTMER. Calls ENEQST, BRNRAT, MGRATE, among others
TBDCALC, TBDZPCALC	Called from DERIV1, GRADNT. Calculates integrals for tapered bore pressure gradient. Also activates printout for TBDZPCALC.
FORMFN	Called from DERIV1. Identifies grain type and sends to specific form function subroutine. Calculates mass fraction burned.
FRMFXN	Turn on every form function for output (where a value of 1 is just initialization information, and 3+ is printed on every call).
FRM07P	Called from FORMFN. 7-perf form function. Will be set to the higher value between it and FORMFN.
FRM19H	Called from FORMFN. 19-perf hex form function. Will be set to the higher value between it and FORMFN.
ENEQST	Called from DERIV1. Calculates updated energy in the chamber, including PRMEAN and TMMEAN. Prints every DERIV1 call.
RESIST	Called from DERIV1. Calculates resistance pressure on projectile.
GRADNT	Called from DERIV1. Calculates pressure at a given location in the chamber. Calculates PRBRCH, PRBASE, among others.
BRNRAT	Called from DERIV1. Calculates burning rate as a function of pressure for each propellant and surface.
HLRATE	Called from DERIV1. Calculates the rate of heat loss.

Table 1. DBUG keywords (continued).

Parameter name	Description
MGRATE	Called from DERIV1. Calculates mass generation rate and other related terms.
VENTIG	Called from MGRATE. Hi-Lo vent gas.
DODPDT	Called from DERIV1. Calculates the rate of change for energy loss.
VENTIS	Called from KUTMER. Hi-Lo vent solid.
UPDATE	Called from KUTMER (via minor subroutine ITSEND). Updates flags after a trajectory output step.
TRJSTORE	Called from KUTMER (via minor subroutine ITSEND). Stores current values into trajectory storage for printing later.
SRIBOP	Called from INTBAL. Subroutine for any optimization routine that only needs to redo the most recent time step (REDOIT [Constant Pressure], REDIT2 [Constant Acceleration Tapered Bore], VCCREDO [VCC Deck]).
REDOIT	Called from SRIBOP. Subroutine for forcing constant pressure calculations.
VCCREDO	Called from SRIBOP. Subroutine for forcing a specific input trace.
PR2VCC2	Called from VCCREDO. Recalculates the PR2 properties for VCC Case=3.
AFTERACTION	Called from INTBAL. Calculates TRAJ values that require a completed simulation. Burning rate reduction for closed bomb output is called from here, and specific vivacity is calculated here as well.
PRINTER	Called from INTBAL. Subroutine that oversees output printing.
PAGCHK	Called from PRINTER. Prints the headers in the main output document. Used to keep track of lines printed to repeat the header on a new sheet of paper but that feature has been mostly disabled.
KMPRN, KMPRN1, KMPRN2, KMPAUX	Called from PRINTER. Controls output printing. Controls multiple subroutines (KMPRN1 [IB simulation], KMPRN2 [Evacuation/Recoil simulation], KMPAUX [TDIS output]).
MODRESET, CLEANUP	Called from main program. Subroutine that clears variables and does some minor housekeeping at the end of the main program.
SETVAR	Deprecated. Sets variable contents by writing miniature input file and reading it. This is how IBHVG1/2/3 operated and has since been replaced. Left in case there is a need for this routine at a later date.

6.3 END Deck Reference

This card has been deprecated. IBHVG looks for the end of file or a SAVE card to begin simulation. If a legacy deck requires an END card, it should run correctly in IBHVG v5. However, if IBHVG seems to be "hanging" while reading the input, try including an END statement and see if that corrects the issue.

6.4 ETC Deck Reference

This deck includes the parameters necessary for simulating an electrothermal-chemical (ETC) propulsion system.[10] Time variables refer to calculated simulation time (0.0 refers to start of IB time). Standard propellant decks may be included as part of the energetic material (Table 2). An example of an ETC simulation is shown in 'test_etc' available in the 'test' folder of the repository.

Table 2. ETC keywords.

Parameter name	Description	Units	Default	Min/Max
NPWR	No. of power-time pairs to describe electric pulse.	. . .	0	2/20
TPWR	Time array of power application.	Seconds	0.0	. . .
PWR	Rate array of power application.	Watts $[in-lb_f/s]$	0.0	. . .
NBRN	Gas generation rate-time pairs.	. . .	0	2/20
TBRN	Time of gas generation pair.	Seconds	0.0	. . .
BRN	Rate of gas generation.	kg/s $[lb_m/s]$	0.0	
NMAS	Mass-time pairs of charge mass % participating in charge/mass (C/M) ratio.	0	2/20	. . .
TMAS	Time array of mass % contribution	Seconds	0.0	. . .
MAS	Charge mass % array in C/M ratio.	Percent	0.0	0.0/1.0

Note: English units and values are included in brackets.

6.5 FIND Deck Reference

The FIND deck features automatic variation of selected input parameters to find a specified IB solution using an algorithm known as the Nelder–Mead or downhill simplex method. This allows a more robust multidimensional operation compared with IBHVG v3 at the expense of run time. Large or complex (multiple FIND decks, or a PARA run with a FIND deck at each iteration) decks may take minutes to run, although typical operation should be under 1 min. Combining a FIND deck with a PMAX deck in the same run is not possible because they rely on completely separate solvers. Instead, consider including an additional FIND deck with a PMAX target value and the same variable that the PMAX deck would vary.

It can be difficult to try to optimize across multiple objectives using multiple find decks. Although the code will run and attempt to achieve a solution, success is less likely, and the Pareto front will not be completely explored. Using an external driver (e.g., Python) to address this scenario will likely result in a better solution.

The solution space is normalized in the code from −1 to 1 across each specified variable. The initial simplex has a point at 0 for all variables (exact mean of each variable range), and then each additional point in the simplex is at 0, except for a single variable with the value of 0.1. VERT allows direct control over the offset for the non-zero simplex vertex in normalized space (−1 to 1). A VERT input whose magnitude is greater than 1 will call a random number instead. **Using VERT in this way might produce different answers from the same input deck each time it is run.** If the FIND routine is not converging, adjust the VERT value to 0.3 or 0.5 as a first course of action.

The FROM value is not used unless a MIN and MAX are not specified. MIN values should be chosen as good runpoints. MAX values will be walked back to find an upper bound, so erring on the side of too much is encouraged. If the MAX value cannot achieve the target, then the FIND routine will exit the run.

EPS controls the tolerance of the given solution. For maximizing/minimizing (CODE=1 or CODE=-1), EPS is the percentage change when moving a vertex. A recommended starting value is 1.001 (a 0.1% change between iterations when moving the vertex). The closer the value is to 1, the more likely it is that the vertex has reached a plateau for the given OUTV target.

For multiple FIND decks, ITER is added across each deck to get the total maximum iterations (Table 3). Fifty to 100 iterations per deck is likely more than enough, but the default value is 200 to try to ensure operation.

It is recommended to set OMIN and OMAX if the variables being changed by the FIND deck are used in their calculation or for maximizing/minimizing in instances in which an a priori bound might exist. See the OUT deck for the default bound equations. However, the defaults are fairly generous, and they may not need to be set at all (e.g., PMAX will default to 2,000 MPa if a value below it is calculated).

Table 3. FIND keywords.

Parameter name	Description	Units	Default value
VARY	Name of parameter in single quotes (' ') plus any array index in ().	. . .	. . .
DECK	Name of deck containing parameter; 4 character maximum, in single quotes (' '). May not be PARA, PDIS, FIND, PMAX, or TDIS.	. . .	. . .
NTH	Number of deck if there is more than one with same name.	. . .	1
FROM (*find_from*)	Starting value of variable in proper units. Not used when MIN and MAX are specified. When no MIN or MAX is specified, it is used to estimate a MIN (FROM*3./5.) and MAX (FROM*5./3.).	. . .	0.0
CODE	**0** to achieve desired value of OUTV variable. **1** to maximize OUTV variable. **-1** to minimize OUTV variable.	. . .	0
OUTV	Name of desired output variable from run completion variables in single quotes (' ') (see OUT deck).	. . .	. . .
EPS	Error tolerance.	CODE 0 OUTV UNITS ELSE %	1.0
TVAL	Desired value of OUTV if CODE=0, else ignored.	. . .	0.0
MULT	Multiplier for function-minimizer residual.	. . .	1.0
MIN (*find_min*)	Minimum-ish allowable value of VARY parameter.	. . .	0.0
MAX (*find_max*)	Maximum-ish allowable value of VARY parameter.	. . .	1.0E+10
ITER (*find_iter*)	Maximum number of iterations for subroutine AMOEBA.	. . .	5000
OMIN	Minimum value of OUTV variable for normalization.	. . .	. . .
OMAX	Maximum value of OUTV variable for normalization.	. . .	. . .
VERT	Normalized location of starting vertex.	. . .	0.1

Note: The python IBHVG class uses the names in parentheses, which include 'find' ('find_from', 'find_min', 'find_max', 'find_iter'). These names do not work in a traditional IBHVG input deck.

6.6 GUN Deck Reference

This deck includes the parameters needed to define the physical aspects of the gun system (Table 4). This deck can activate additional pressure output that is useful for comparing the model output with a pressure tap from a specific gun barrel.

Table 4. GUN keywords.

Parameter name	Description	Units	Default
NAME, TYPE	Name of gun system; alphanumeric, up to 28 characters.	. . .	. . .
CHAM, CV, VOL	Chamber volume.	m^3 [in^3]	0.0
GRVE	Groove diameter (see Figure 4).	m^3 [in^3]	0.0
LAND	Land diameter (see Figure 4).	m^3 [in^3]	0.0
TRAV, LENG	Travel to shot exit.	m [in]	0.0
G/L, *G2L*	Groove-to-land ratio.	. . .	0.0
TWST	Rifling twist.	calibers/turn	999.0
LDEN	Ratio of total charge weight to chamber volume; used when LOPT is 1 or 2.	kg/m^3 [lb_m/in^3]	0.2
LOPT	0 = ignore LDEN 1 = calculate CHAM from total charge weight and LDEN 2 = calculate charge weight from CHAM and LDEN Primer and charge weights must be specified; they are scaled proportionately to sum to required total charge weight.	. . .	0
NGAG	Number of gage locations.	. . .	0
GLOC	Gage location array of size NGAG distance is measured from the breech face if positive, or into the chamber from the projectile base if negative.	m [in]	0.0
CLEN	Effective chamber length for scaling in-chamber gage locations.	m [in]	CV/bore area
CPTS	Number of chamber/gun diameter positions as measured from breech face.	m [in]	0
DIST	Distance measured from breech face for corresponding chamber/gun diameter; CPTS pairs of points.	m [in]	0.0
DIAM, ID, BORE	Interior diameter of chamber/gun at corresponding DIST from breech.	m [in]	0.0
OD	Exterior diameter of chamber/gun at corresponding DIST from breech.	m [in]	0.0
VOLB	Volume added to chamber when telescoping chamber option in effect added when PROP deck IGNC=5 or 6 and THRC condition is met.	m^3 [in^3]	0.0

Note: The python IBHVG class uses the parameter names in italics. English units and values are included in brackets.

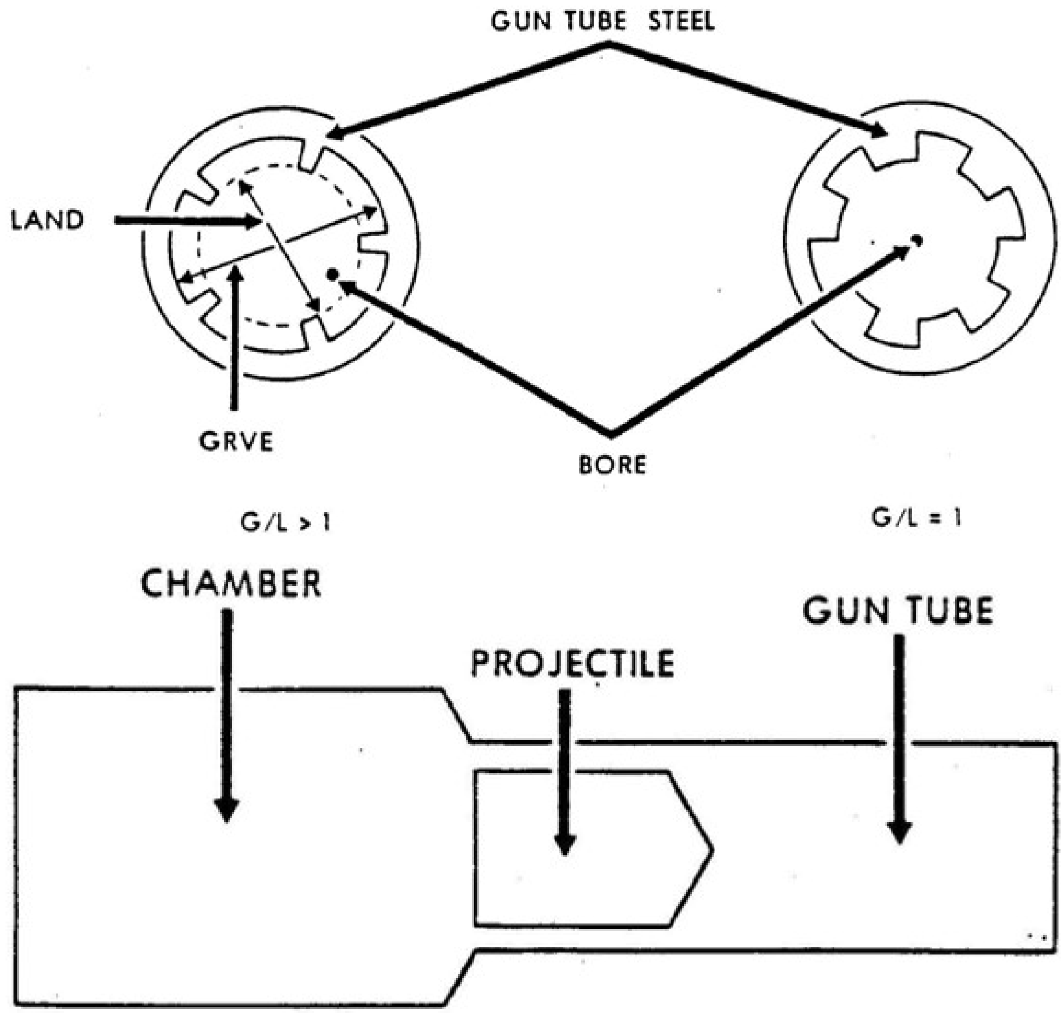

Figure 4. Diagram of a simplified gun system with keywords and concepts shown.[1]

6.7 HEAT Deck Reference

For heat-loss-related variables; heat energy transferred from hot combustion gases to a thin shell of gun steel on the inner walls of chamber and tube (Table 5).

Table 5. HEAT keywords.

Parameter name	Description	Units	Default
TSHL	Tube shell thickness for heat sink.	m [in]	0.000102 [0.004]
CSHL	Shell-specific heat covering a broad range of steels.	$\frac{J}{kg \star K}$ $[\frac{in-lb_f}{lb_m \star K}]$	461.80 [1848.0]
RSHL	Density of shell material.	kg/m^3 $[lb_m/in^3]$	7861. [0.284]
TWAL	Initial wall temperature K.	[K]	293.0
H0	Free convective heat transfer coefficient for air in the tube.	$\frac{W}{m * K}$ $[\frac{in-lb_f}{in^2 \star s \star K}]$	11.348 [0.0648]
HL	**0** = ignore heat losses in energy balance **1** = include heat losses	. . .	1
STAN	Static N value for specific vivacity TRAJ variable calculation.	. . .	1

Note: English units and values are included in brackets.

6.8 INFO Deck Reference

The INFO Deck contains run-related input and output options. The numerical operation inputs are listed in Table 6, and input and output options are given in Table 7.

Table 6. INFO keywords related to numerical operation.

Parameter name	Description	Units	Default
EPS	Maximum error for integrator time-step adjustment and transition tolerances.	. . .	0.002
DELT	Maximum integration time step.	s	0.0001
DELP	Integrator logout and print step reset to DELT if DLPU=1 and DELP<DELT.	s or m [s or in]	. . .
DLPU	**1** for DELP in units of time. **2** for DELP in units of projectile travel.	s [s] m [in]	1
LOOP	Maximum loop iterations for the KUTMER subroutines. Resets on each entry to the subroutine. Examples: each run of a PARA search or each entry from the VCC subroutine (which enters and exits to check to see if it achieves the desired value) is subject to its own maximum iterations. **0** allows infinite iterations (backwards compatible). Negative numbers exit with an error, positive values exit KUTMER but will continue operation.	. . .	0

Table 7. INFO keywords related to I/O and external programs.

Parameter name	Description	Units	Default
RUN, TITL	Run title printed on page headers. 48 character maximum.	. . .	. . .
INP	Choice for input values. **1** for English units. Any other option is metric units.	. . .	0
OUT(1)	Choice of units for standard output file. **1** for English units. Any other option is metric units.	. . .	0
OUT(2)	**1** create copy of input decks in File 12 using alternate units.	. . .	0
GNUP	String of path to gnuplot executable, if required. Defaults to gnuplot installation folder in IBHVG executable directory.	. . .	*'./gnuplot/bin/gnuplot.exe'*

Note: The python IBHVG class uses the parameter names in italics. English units and values are included in brackets.

Parameter **POPT** (Print OPTions): an array of size 10 is explained in Table 8.

Table 8. INFO parameter POPT options.

POPT index	Description	Default
1	**0** Suppress input print echo. **1** Print input data echo.	1
2	**0** Suppress trajectory print. **1** Print default trajectory variables. **2** Print user-specified TDIS variables. **3** Print only POPT(9), POPT(10), or Plot (to their respective separate files).	1
3	**0** Suppress IB summary. **1** Print IB summary information.	1
4	**0** Suppress tube blowdown calculation. **1** Print blowdown phase to include tube recoil if recoil option is in effect. **2** Print blowdown phase until breech pressure reaches BLPR value.	1
5	**0** Honor above print options for every run of parametric variation. **1** Honor above print options for first run of parametric variations, then print a single-line summary for the rest. **2** Like option 1, but use print variables supplied via PDIS decks.	1
6	RESERVED	1
7	RESERVED	. . .
8	RESERVED	. . .
9	**0** Suppress any PARA .csv file generation. **1** Print the selected varied inputs and all output OUT variables to a separate .csv file (e.g., machine operation/parametric operation). **2** Print user-specified PDIS variables to separate .csv file.	0
10	**0** Suppress any TRAJ .csv file generation. **1** Print all TRAJ variables to separate .csv file (e.g., machine operation/parametric operation). **2** Print user-specified TDIS variables to separate .csv file.	0

For PARA, POPT(10) options **1** and **2** will produce multiple files with the parametric loop number appended to the file name. This is the same number used for PLOTs that are generated during use of a PARA deck. Keywords related to pressure options are in Table 9, and all options for constant pressure gun simulation are in Table 10.

Table 9. INFO keywords related to IB simulation.

Parameter name	Description	Units	Default
GRAD	**1** for Lagrangian pressure gradient. **2** for Pidduck–Kent gradient. **3** for chambrage gradient (projectile starts at end of GUN DIST/DIAM values). **4** for Tapered Bore gradient (GUN DIST/DIAM values must extend to muzzle, CHAM determines projectile start location by integration of gun tube profile).	. . .	1
BLPR	Target value for breech pressure if blowdown option [POPT(4) - see above] is 2.	MPa [psi]	0.0 [0.0]
CPTB	**0** Default IB simulation. **1** With Tapered Bore gradient (GRAD=4), optimize bore diameter for a constant base pressure-tapered section.	. . .	0

Note: English units and values are included in brackets.

Table 10. INFO keywords related to constant pressure gun analysis options.

Parameter name	Description	Units	Default
CONP	**0** Standard IB calculation with pressures allowed to vary due to combustion gas creation. **1** Maintain constant pressure by varying the burning rate of a single charge (requires maximum of one PROP deck). **2** Maintain constant pressure by varying surface area of single charge (same PROP restrictions).	. . .	0
PRES	Desired constant pressure if CONP>0.	MPa [psi]	0.0 [0.0]
TOL	Error tolerance for PRES if CONP>0.	MPa [psi]	0.0001 [0.6895]

Note: English units and values are included in brackets.

6.9 OUT Deck Reference

The OUT Deck gives a list of parameters that are available at the end of IB calculations for printing with PDIS or as search variables with the FIND deck (Table 11).

The default minimum and maximum values are used for normalization in the FIND routine and can be overridden from there. PMAX's upper bound is calculated or 2,000 MPa, whichever is greater.

Table 11. **OUT variable names and definitions.**

Parameter name	Description	Units	Default MIN	Default MAX
PMAX	Maximum breech pressure.	MPa [psi]	0.101325	Calculated using CHWT and initial CHAM values
HUMP	Array of size 5 of local breech pressure maxima.	MPa [psi]	0.101325	Calculated using CHWT and initial CHAM values
GMAX	Array of size 30 of gage pressure maxima.	MPa [psi]	0.101325	Calculated using CHWT and initial CHAM values
VMUZ	Muzzle exit velocity of projectile.	m/s [ft/s]	0.0	$\mathrm{sqrt}\left(\frac{2 \star C \star I}{m_{projectile} \star (\gamma - 1)}\right)$
AMAX	Maximum acceleration.	g [g]	0.0	$\frac{PMAX_{max} \star A_{bore}}{m_{projectile}}$
BMAX	Maximum projectile base pressure.	MPa [psi]	0.101325	Calculated using CHWT and initial CHAM values
X@BO	Array of size 40 of projectile position at charge burnouts referenced by subscript.	m [in]	0.0	CLEN+TRAV
PMUZ	Projectile base pressure at shot exit.	MPa [psi]	0.101325	Calculated using CHWT and initial CHAM values
ZMUZ	Array of size 40 of charge weight fractions burned at shot exit referenced by subscript.	%	0.0	1.0
IMPL	Final impulse (momentum)	$N \star s$ [$lb_f \star s$]	0.0	$m_{recoil} \star \mathrm{sqrt}\left(\frac{2 \star C \star I}{m_{recoil} \star (\gamma - 1)}\right)$
LDEN	Calculated loading density	g/cm^3	0.0	Mass averaged RHO

Note: English units and values are included in brackets.

6.10 PARA Deck Reference

The PARA deck is used for parametric variations of different variables. The PARA deck must contain a nominal value of the variable to be varied (Table 12).

Recommended to use POPT(2)=3, POPT(9)=1, and POPT(10)=1 from the INFO if more than a trivial (single dimension, limited number of steps) use case. These options print each run to a separate .csv file and generate a .csv file with the corresponding variables being adjusted and all of the output variables.

Table 12. **PARA keywords.**

Parameter name	Description
VARY	Name of parametric variable in single quotes (' ') plus any array index in ().
DECK	Name of deck containing variable; 4 character maximum, in single quotes (' '). May not be PARA, PDIS, FIND, PMAX, or TDIS.
NTH	Deck number if there is more than one such deck (*default = 1*).
FROM (*para_from*)	Initial value of variable.
TO	Final value of variable (variation will not exceed this value).
BY	Increment/decrement value.

Note: The python IBHVG class uses the names in parentheses, which include 'para' ('para_from'). These names do not work in a traditional IBHVG input deck.

6.11 PDIS Deck Reference

Each PDIS deck defines one variable to be printed in lieu of the default set for each line of the parametric summary print (Table 13). Be sure to include the line POPT(5)=2 (print option) in the INFO deck. Optional variables to be printed can also be found in the OUT and TRAJ decks.

Table 13. PDIS keywords.

Parameter name	Description	Default
SHOW	Name of variable to print in single quotes (' '). 4 characters maximum plus any array index in ().	. . .
DECK	Name of deck containing the desired variable in single quotes (' ') any deck may be named except PARA, PDIS, FIND, PMAX, and TDIS.	. . .
NTH	Number of deck if more than one 1.	. . .
MULT	Number by which to multiply data.	1.0
DIV	Number by which to divide data.	1.0
REM1, REMK	Remark string for titles (up to 20 characters) in single quotes (' ').	. . .

Note: The python IBHVG class uses the parameter names in italics.

6.12 PLOT Deck Reference

Creates plots using gnuplot. Gnuplot does not need to be installed, but it does need to be present. Simply download the compiled binary in .7z format from their website and unzip it or copy it from an existing installation. Every PLOT deck gives a new .png image file (Table 14). The filename will be the same as the text output filename with either the LABL string appended (if entered) or the PLOT deck number (1, 2, etc.) appended. The data output file for gnuplot is PDATA.dat, if you overwrite the defaults with the FILE command. Six line styles are included by default. If more than 6 lines are wanted on a plot, overwriting the command file or inputting additional commands is recommended. If multiple Y variables are plotted, the y-axis is titled with the units of the first variable and legend with each variable name is included. When using a PARA deck, the filenames will be appended with the loop iteration number to prevent overwriting during the parametric runs. This number matches the POPT(10) number in INFO.

Issues may occur with this function and the Intel Fortran Compiler.

Table 14. PLOT keywords.

Parameter name	Description	Default
LABL	Title string of plot, will also append to filename.	. . .
FILE	String of gnuplot input file name for more complicated plots. **Overrides ALL defaults.**	PLOTinput.inp
NX1	Size of X variable array.	1
NY1	Size of Y variable array.	1
NY2	Size of Y2 variable array.	0
XVAR	Array of strings denoting the X variable name array.	. . .
YVAR	Array of strings denoting the Y variable name array.	. . .
Y2VA	Array of strings denoting the Y2 variable name array.	. . .
XDEK	Array of strings denoting the deck the corresponding X variable name is found in TRAJ	…
YDEK	Array of strings denoting the deck the corresponding Y variable name is found in TRAJ	…
Y2DK	Array of strings denoting the deck the corresponding Y2 variable name is found in TRAJ	…
XNTH	Array of values denoting the deck number to read for XDEK (similar to NTH in TDIS).	0
YNTH	Array of values denoting the deck number to read for YDEK (similar to NTH in TDIS).	0
Y2NT	Array of values denoting the deck number to read for Y2DK (similar to NTH in TDIS).	0
XR1	Array of 2 values denoting the lower and upper bounds of the X-axis. No entry lets gnuplot autoscale the axis.	. . .
YR1	Array of 2 values denoting the lower and upper bounds of the Y-axis. No entry lets gnuplot autoscale the axis.	. . .
YR2	Array of 2 values denoting the lower and upper bounds of the Y2-axis. No entry lets gnuplot autoscale the axis.	. . .
SCX	Array of scale factors for X variables. If a negative value is specified, the X variable is normalized by the maximum value of the variable specified by the index. (e.g., -1 normalizes by the maximum value of the 2nd value in XVAR). The default value of 0 does not scale values.	SCX

Table 14. PLOT keywords (continued).

Parameter name	Description	Default
SCY	Array of scale factors for Y variables. If a negative value is specified, the Y variable is normalized by the maximum value of the variable specified by the index. (e.g., -1 normalizes by the maximum value of the 2nd value in XVAR). The default value of 0 does not scale values.	0
SCY2	Array of scale factors for Y2 variables. If a negative value is specified the Y2 variable is normalized by the maximum value of the variable specified by the index. (e.g., -1 normalizes by the maximum value of the 2nd value in XVAR). The default value of 0 does not scale values.	0
NCOM	Size of the command array	0
COMS	Array of strings to add extra GNUPLOT command lines to the plot creation file.	. . .

Some useful extra commands that can be input via COMS are as follows:

```
set logscale xy
```

```
set grid x y
```

These are particularly useful for burn rate plots by changing to a logarithmic scale in x and y and adding grid lines.

For normalization and multiple Y datasets, it is recommended to include a COMS command to rewrite the x or y label.

```
set xlabel "<label>" font " , 30" offset 0,-5
```

```
set ylabel "<label>" font " , 30" offset -14,0
```

These are examples where the `" , 30"` adjusts the font size and the offset value moves the text out of the axes numbering (note the –y offset for the x-axis and the –x offset for the y-axis).

More info on what can be entered via the COMS command or overwritten with the FILE command can be found in the GNUPLOT manual.

Default Plot File:

```
set term png truecolor gian size 1600,1200
set output *filename*.png
set border lw 4
set style line 1 lw 4 lc rgb "blue"
set style line 2 lw 4 lc rgb "red"
set style line 3 lw 4 lc rgb "green"
```

```
set style line 4 lw 4 lc rgb "magenta"
set style line 5 lw 4 lc rgb "cyan"
set style line 6 lw 4 lc rgb "yellow"
set style line 101 lw 3 lc rgb "gray"
set style line 102 lw 1 lc rgb "gray"
set style line 103 lw 3 lc rgb "black"
**if NY1=1** set key off **else** set key inside left top Left fo
nt " , 30"
set xrange [*XR1(1)*:*XR1(2)*]
set yrange [*YR1(1)*:*YR1(2)*]
**if NY2>0** set y2range [*YR2(1)*:*YR2(2)*]
set margins 30, 30, 10, 5
set xlabel *label* font " , 30" offset 0,-5
set xtics offset 0,-2
set ylabel *label* font " , 30" offset -14,0
**if NY2>0** set y2label *label* font " , 30" offset 14,0
set tics nomirror scale 2.5 font " , 30"
**if NY2>0** set tics nomirror scale 2.5 font " , 30"
set grid x y
set grid mx my ls 101, ls 102
set style data lines
```

6.13 PMAX Deck Reference

The PMAX Deck is used for variation of charge weight or web to achieve a desired maximum breech pressure (Table 15). The PROP deck must contain a nominal value of charge weight and web, even though one or the other will be varied in the PMAX operation. Since PMAX knows it will be varying a propellant parameter, no DECK='PROP' line is required (like that for FIND or PARA).

A relative tolerance of 1e-6 is included by default, which is defined by mean=(Guess(2)+Guess(1))*0.5, rtol = abs(Result(2)-mean/mean)<1.e-6.

After the maximum pressure has been bracketed, a slope between trial points is calculated and used to identify a new trial point. In the case of repeated trial points below 5% or above 95% of the span, a bisection will occur instead, in an attempt to get a more accurate slope.

Table 15. PMAX keywords.

Parameter name	Description	Units	Default
ON	Switch for turning off PMAX decks in subsequent runs (e.g., when used with SAVE). If VARY is inputted, ON is set to 3 (i.e., ON - for programmatic reasons) to maintain legacy input operation. Otherwise the default is 0. Setting ON = 0 will turn off PMAX operation.		3
VARY	Variable name from a PROP deck, including array index (e.g., WEB(2))		
NTH	Number of PROP deck (1 = first deck, etc.)		1
TRY1	First value of varying parameter to try		
TRY2	Second value to try; third and subsequent guesses are based on interpolation. NOTE: the last two guesses from the previous run are employed as the first two guesses in the second and subsequent PARA runs		
PMAX	Maximum breech pressure sought	MPa [psi]	
XEPS	Tolerance on the chosen input variable	dependent on VARY	0.0
YEPS EPS	Error tolerance for pressure during PMAX run	MPa [psi]	0.0069 [1.0]
LOOP	Maximum number of tries before quitting		20
MIN (pmax_min)	Minimum allowable value of VARY parameter		0.0
MAX (pmax_max)	Maximum allowable value of VARY parameter		1.E+10
NPMX	0 if PMAX refers to max breech pressure during run n where n>0 and n<6 if PMAX refers to the NPMX-th local breech pressure maximum achieved		0
PRNT	If PRNT>0, print an intermediate line during the computation. Shows current iteration number, the VARY parameter, its last value and the corresponding $PMAX, and the next value to be tried		0

Note: The python IBHVG class uses the names in parentheses which include 'pmax' ('pmax_min', 'pmax_max'). These names do not work in a traditional IBHVG input deck. English units and values are included in brackets.

6.14 PRIM Deck Reference

Primer data describes propellant completely burned at the start of the IB calculation, providing a "jump start" for pressure-dependent propellant burning rates from the PROP decks. Usually, it is best to only use a portion of the real primer in the PRIM deck, enough to ignite the other propellants, while the rest of the primer charge may be modeled as an accompanying PROP deck. Using the actual total primer mass in the PRIM deck often generates too much pressure in the chamber and results in an overpowering jolt to the burning rates and projectile motion. Constant-pressure runs may include or exclude a primer, but conventional simulations require a PRIM deck (Table 16).

Table 16. PRIM keywords.

Parameter name	Description	Units
NAME, TYPE	Name of primer in single quotes (' '). 28 characters maximum.	. . .
GAMA	Primer material-specific heat ratio.	. . .
FORC	Force of the primer material.	J/kg $[\mathrm{ft-lb_f/lb_m}]$
COV	Covolume of primer propellant.	$\mathrm{m^3/kg}$ $[\mathrm{in^3/lb_m}]$
TEMP	Flame temperature.	K
CHWT, WT, CW, CHGW, C	Primer mass.	kg $[\mathrm{lb_m}]$

Note: The python IBHVG class uses the parameter names in italics. English units and values are included in brackets.

6.15 PROJ Deck Reference

The PROJ deck contains projectile-related variables (Table 17).

Table 17. PROJ keywords.

Parameter name	Description	Units
NAME, TYPE	Projectile designation. 28 characters maximum.	. . .
PRWT, WT	Projectile mass.	kg $[\mathrm{lb_m}]$
COPT	**0** ignore C/M (default). **1** calculate PRWT from total charge mass and C/M. **2** calculate total charge masses from PRWT and C/M; primer and all charge weights must be specified, but they will be scaled proportionately to sum to the required total mass.	. . .
C/M, *C2M*	Used when COPT=1 or 2. Ratio of total charge mass and projectile mass (default=1.0).	. . .
SOPT	**0** bypass PRWT calculation based on subprojectile parameters and sabot formula (default). **1** find PRWT based on projectile weight estimation formula of Burns.[11] **2** find PRWT based on same weight estimation formula using coefficients yielding 15% lighter sabot.	. . .
WTSP	Subprojectile weight.	kg $[\mathrm{lb_m}]$
LSP	Subprojectile length.	m [in]
DSP	Subprojectile diameter.	m [in]
PDES	Maximum design pressure for saboted projectile.	MPa [psi]
SABO	Sabot weight (not an input—set by the program).	kg $[\mathrm{lb_m}]$

Note: The python IBHVG class uses the parameter names in italics. English units and values are included in brackets.

6.16 PROP Deck Reference

The PROP deck contains the variables needed to describe a single propellant. Nominal thermodynamic values for many military propellants are found in the work by Freedman.[12] The high-level variables needed to define a propellant are included in Table 18.

Table 18. PROP keywords: general variables.

Parameter name	Description	Units	Default
NAME, TYPE	Name of propellant; 28 characters maximum.	. . .	. . .
CHWT, WT, C, CHGW, CW	Charge weight.	kg [lbm]	0.0 [0.0]
NTH	Deck number of a different PROP deck with which to link properties. Additional properties listed in this PROP deck will remain unlinked (e.g., same propellant but portions delayed using IGNC and THRC; allows optimization of burn rate parameters across all linked PROP decks). If NLYR is adjusted, it must be adjusted individually for ALL propellants linked to this one, even if no layered properties are adjusted.	. . .	. . .
GRAN, FORM	Propellant granulation code chosen from 7PF (or 7P), 1PF (or 1P), CORD, RECT (SLAB), SPHR (BALL), SLOT, 37HX (37H), 19HX (19H), 19PF (19P), GEN, PIE (STAR), GHEX (HEX), MONO, PAN (CAKE), ROSE, SPFF (SPF2), CIGA.	. . .	. . .
NLYR	Number of layer surfaces for all data.	. . .	4

Note: The python IBHVG class uses the parameter names in italics. English units and values are included in brackets.

The thermochemical and burn rate data can be entered as uniform values (for the propellant) by using the keywords in Table 19. More detailed ways to define propellant properties can be found starting with the keywords in the table on page 47.

Table 19. PROP keywords: uniform thermochemical information and burn rate.

Parameter name	Description	Units	Default
RHO, DENS	Density of propellant material	kg/m^3 [lbm/in^3]	. . .
GAMA	Specific heat ratio	. . .	. . .
FORC	Force	J/kg [ft-lbf/lbm]	. . .
COV	Covolume	m^3/kg [in^3/lbm]	. . .
TEMP	Flame temperature	K	. . .
ALPH	Burning rate exponent (α), where burning rate = βP^{α} and P is mean pressure.	m/s [in/s]	. . .
BETA	Burning rate coefficient (β), where burning rate = βP^{α} and P is mean pressure.	$\frac{\mathrm{m/s}}{\mathrm{MPa}^{\alpha}}$ [$\frac{\mathrm{in/s}}{\mathrm{psi}^{\alpha}}$]	. . .
EROS	Erosive burning coefficient—empirical factor multiplied by projectile velocity (m/s) to add to burning rate.	. . .	0.0

Note: The python IBHVG class uses the parameter names in italics. English units and values are included in brackets.

Available form functions to describe the volume and surface area of the propellant are given in Table 20. The parameters required to define a given form function are included, and visuals of the form functions and their parameters are shown in Figures 5–7. Some propellants allow multiple measurements that can overdefine the geometry. In these cases, the WEB typically takes precedence over the DIAM of the propellant, and a warning should be printed that this is occurring by the code, but it is wise to double check the input data echo.

Table 20. PROP keywords: form functions.

Form function name	Description	Required parameters
7PF, 7P	Cylindrical grain with seven lengthwise perforations[13]	LEN, PD, WEB (or DIAM) uses ratios: WI/WO, D/PD, L/D, L/PD
1PF, 1P	Cylindrical grain with one lengthwise perforation[13]	LEN, PD, WEB (or DIAM) uses ratios: WI/WO, D/PD, L/D, L/PD
CORD	Cylindrical with no perfs[13]	LEN, DIAM uses ratios: L/D
RECT, SLAB	Rectangular solid[13]	LEN, WDTH, THCK uses ratios: WD/T, L/WD
SPHR, BALL	Spherical solid grain[13]	DIAM uses ratios: None
SLOT	Slotted single-perf grain[13]	LEN, PD, WEB (or DIAM), SLOT uses ratios: WI/WO, D/PD, L/D, L/PD, SW/D

[a] Form functions have been added since the original IBHVG2 report.[1]

Table 20. PROP keywords: form functions (continued).

Form function name	Description	Required parameters
37HX, 37H	37-perf hexagonal grain, perfs arrayed (not ringed)[1]	LEN, PD, WEB (or DIAM) uses ratios: WI/WO, D/PD, L/D, L/PD
19HX, 19H	19-perf hexagonal grain, perfs arrayed (not ringed)[1]	LEN, PD, WEB (or DIAM) uses ratios: WI/WO, D/PD, L/D, L/PD
19PF, 19P	19-perf cylindrical grain, perfs arrayed (not ringed)[13]	LEN, PD, WEB (or DIAM) uses ratios: WI/WO, D/PD, L/D, L/PD
GEN	Generalized grain with measurements of surface area .vs depth burned. Last array values will be maintained if charge mass is not exhausted by then (or if 0 maintained and will prevent mass exhaustion).[14]	NSUR (number of array pairs), DEPB, SURF uses ratios: None
PIE, STAR	Cylindrical grain with lengthwise interior slots[1]	LEN, DIAM, SLOT, NSLT uses ratios: L/D, L/PD, SW/D
GHEX, HEX	General hexagonal grain with specified number of perforation rings[1]	LEN, PD, WEB, NRNG uses ratios: WI/WO, D/PD, L/D, L/PD
MONO	Monolithic cylindrical grain with lengthwise perforations[1] (no. of perfs to be determined)	LEN, PD, WEB, DIAM uses ratios: None
PAN, CAKE[a]	Flattened ball grain having defined thickness and diam	THCK, DIAM (or WRED and ORGD) uses ratios: None
ROSE[a]	Rosette grain with defined number of lengthwise perforation rings.[15] Perforations arranged similar to a HEX grain	LEN, PD, WEB, NRNG uses ratios: WI/WO, D/PD, L/D, L/PD
SPF2, SPFF[a]	Single-perf grain burning only on inner perf surface Perforations arranged similar to the GHEX grain Propellant layer properties required on P surfaces	LEN, PD, WEB (or DIAM) uses ratios: WI/WO, D/PD, L/D, L/PD
CIGA[a]	Cord grain only burning at one end (cigarette-style) Propellant layer properties required on E surfaces	LEN, DIAM (or DFWD, DAFT; forward and aft diameters of a frustrum) uses ratios: L/D
QLOD[a]	QuickLOAD[16] Compatible Form Function	QLA0, QLBA, QLB, QLZ1 uses ratios: None

[a] Form functions have been added since the original IBHVG2 report.[1]

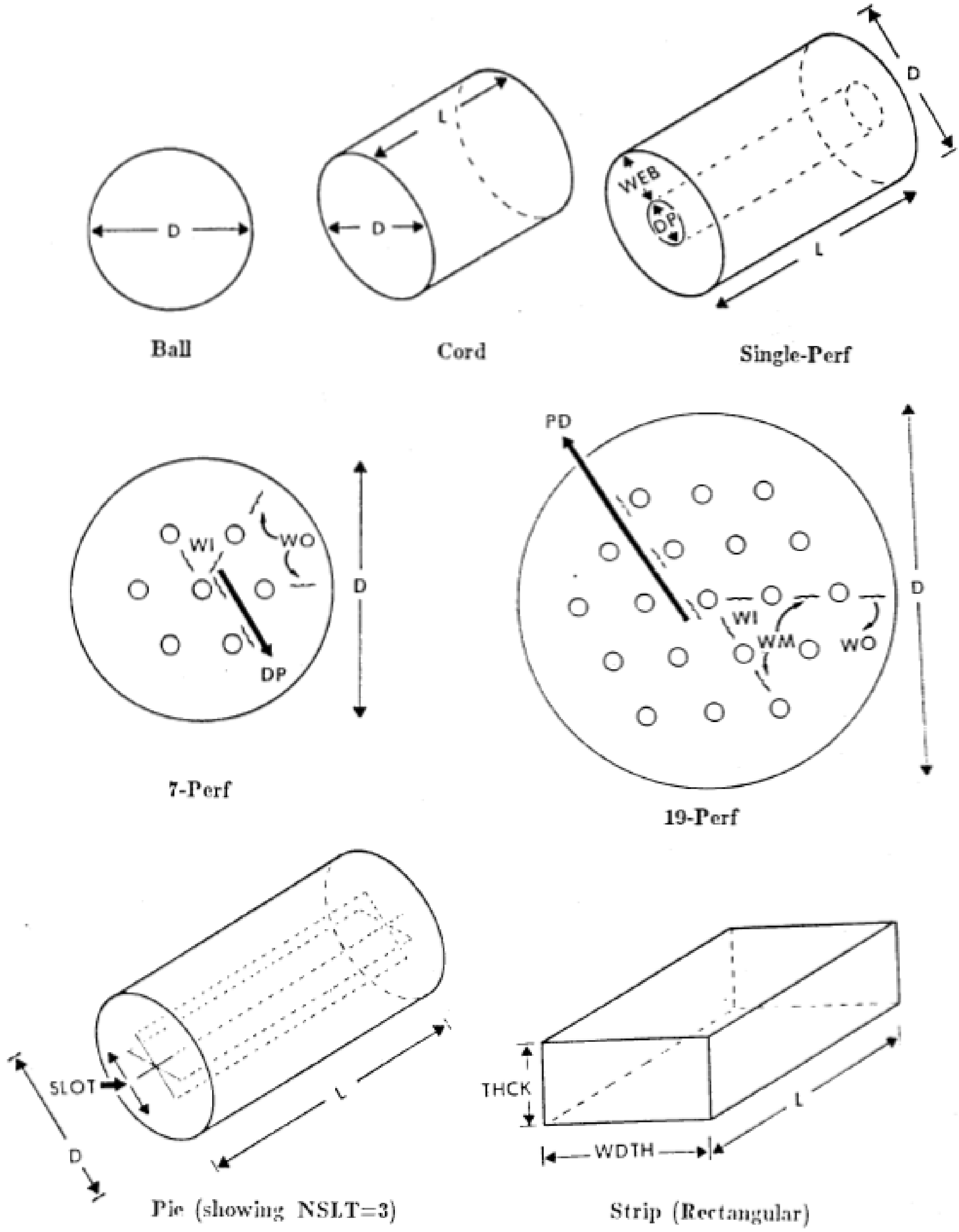

Figure 5. Grain types and parameters.[1]

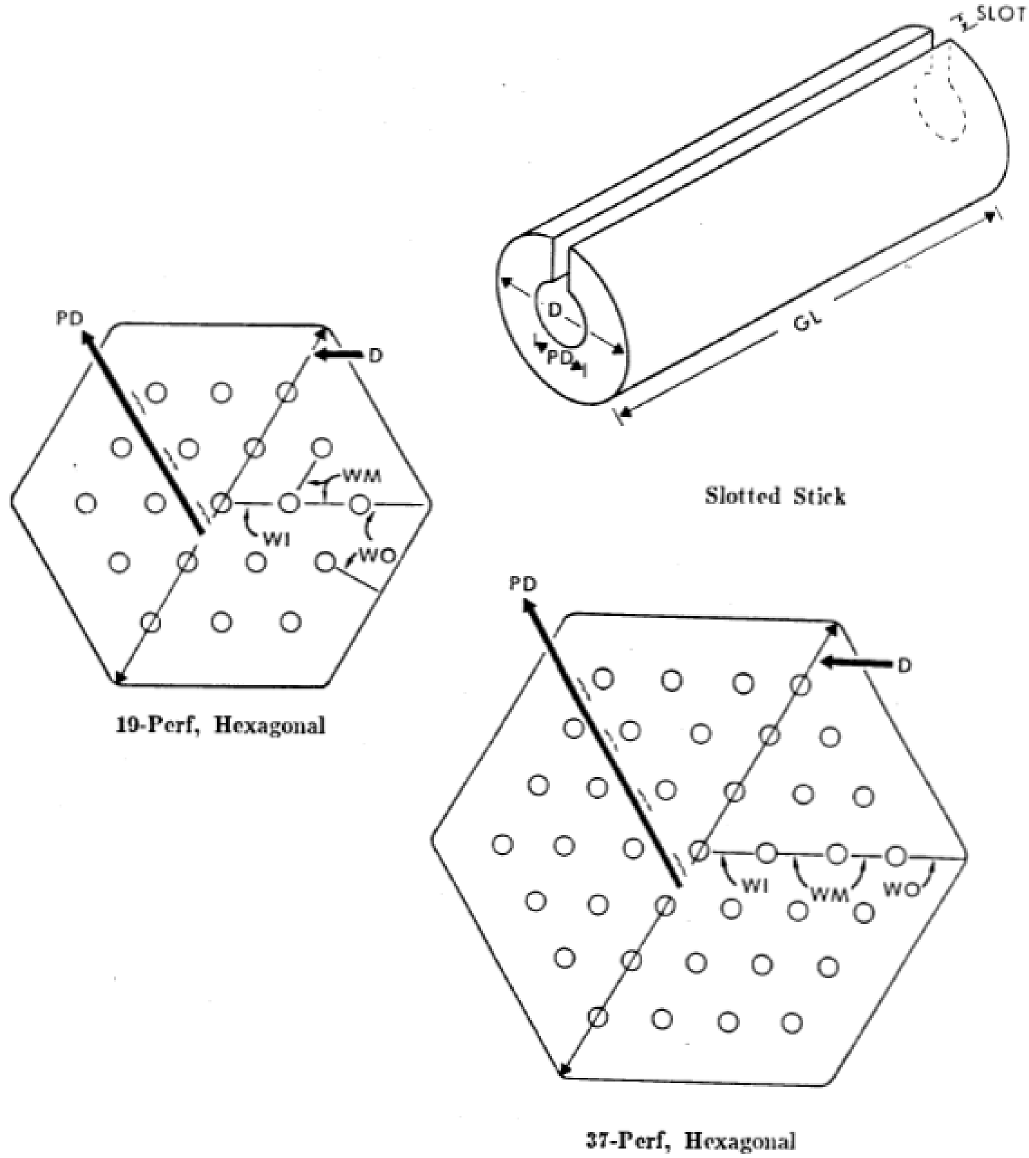

Figure 6. Additional grain types and parameters.[1]

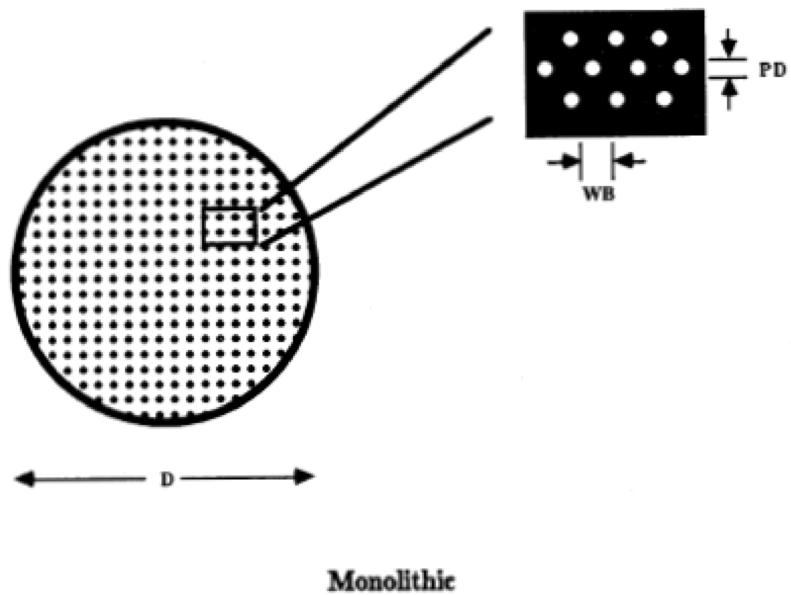

Figure 7. The monolithic grain type and its parameters.[1]

The QuickLOAD compatible form function requires its own special parameters (Table 21). For QuickLOAD users, the names might make more sense without the QL preceding the key word (i.e., QLB is equivalent to b) and should allow direct copying over. The non-required variables are included to make the form function more flexible for the IB modeler if needed. Exact matches between QuickLOAD and IBHVG might not occur because of differences in how they handle the thermochemical description of the propellant (i.e., QuickLOAD's use of Heat of Explosion) among other differences. The inclusion of this form function does not endorse or otherwise express any opinion, positive or negative, about the QuickLOAD software.

Table 21. PROP keywords: QuickLOAD form function.

Parameter name	Description	Units	Default
QLN	Progressive region exponent (not burn rate exponent).		0.5
QLM	Regressive region exponent (not burn rate exponent).		0.5
QLA0	Progressive region mass fraction coefficient. QuickLOAD variable name: a0 (Located under Pro-/Degressivity Coefficient)		. . .
QLA1	Regressive region mass fraction coefficient. -1.0 enforces a surface area of 0 at a mass fraction of 1.		-1.0
QLB	Regressive region coefficient. QuickLOAD variable name: b (Located under Coefficients of Arcs)		. . .
QLBA	Initial Surface Area * Burning Rate Coefficient (used as pure Burning Rate Coefficient). QuickLOAD variable name: Ba (Located under Coefficient of Arcs)		. . .
QLRA	Ratio of QLBA for burning rate vs. surface area. SA∝QLRA, R_b∝QLBA/QLRA.		1.0
QLEX	Burn rate exponent.		1.0
QLZ1	Transition mass fraction from progressive to regressive region (e.g., 0.45). QuickLOAD variable name: Limit of z1 (P/Pmax or z – Located under Intersection of Both Arcs)		. . .
QLP0	Reference pressure. For legacy operation, reference pressure is 1 bar.		1.e5

The grain dimensions are defined in Table 22.

Table 22. PROP keywords: grain dimensions.

Parameter name	Description	Units	Default
WI, WIN	Inner web for cylindrical/hexagonal grains.	m [in]	0.0
WO, WOUT	Outer web for cylindrical/hexagonal grains.	m [in]	0.0
WM, WMID	Middle web for cylindrical/hexagonal grains.	m [in]	0.0
D, DIAM, GDIA	Grain diameter, if applicable. For HEX grains (e.g., is 19 perfs), this is still $D = 5 \star pd + 6 \star web$ (i.e., measured from point to point, not across the flats). $flats = D * cos(30°)$).	m [in]	0.0
PD, DP, PDIA	Perforation diameter, if applicable.	m [in]	0.0
SLOT	Slot width in SLOT and PIE grains.	m [in]	0.0
NSLT	Number of diametral slots in PIE grain (>1).		0
NRNG	Number of concentric rings of perforations around central perf in HEX and ROSE grains.		2
L, GLEN, LEN, GL	Grain length, if applicable.	m [in]	0.0
WDTH	Grain width in RECT grains.	m [in]	0.0
THCK	Grain thickness in RECT and PAN grains.	m [in]	0.0
ORGD	Original diameter for PAN grains. Alternative to a negative DIAM with PAN for use with WRED.	m [in]	0.0
WRED	ORGD reduction for PAN grains THCK = ORGD*(1-WRED).	m [in]	0.0
DFWD	Diameter of burning end of CIGA grain.	m [in]	0.0
DAFT	Diameter of non-burning end of CIGA grain.	m [in]	0.0
SMLT	Burning surface area multiplier for CIGA grain.		. . .
NSUR	Number of depth/surface pairs for GEN grain.		1
DEPB	Depths-burned array for GEN grains; ignored if NSUR = 1.	m [in]	0.0
SURF	Surface area array for GEN grains.	m^2 [in^2]	0.0
GRNS	Number of grains for GEN form function.		1

Note: The python IBHVG class uses the parameter names in italics. English units and values are included in brackets.

For some of the grain types, the code will calculate grain dimensions from a given measurement and ratio(s) to the remaining lengths. This feature can be quite useful when trying to optimize a charge design. For example, a CORD grain can be defined by D and L, or by L and the ratio of L/D. Not all grain types make use of this feature, and the experienced modeler will carefully check the output before relying on this function, particularly when using a PARA deck. (Recommended spot-checking via POPT(5)=0 in INFO.) In general, an inputted ratio will take precedence over any given dimension (e.g., if DIAM and D/DP are inputted, the DIAM value will be ignored). This is also true for WEB (e.g., if WEB and D/DP are inputted, the WEB will be ignored because the ratio takes precedence). The next series of PROP deck inputs are used for automatic grain dimension calculations using ratios to other values (Table 23).

Table 23. PROP keywords: grain ratios.

Parameter name	Description	Default
L/D, *L2D*	Grain length/grain diameter	0.0
L/DP, L/PD, *L2DP*, L2PD	Grain length/grain perforation diameter	0.0
D/DP, D/PD, *D2DP*, D2PD	Grain diameter/grain perforation diameter	0.0
L/WD, *L2WD*	Grain length/grain width	0.0
WD/T, *WD2T*	Grain width/grain thickness	0.0
SW/D, *SW2D*	Slot width/grain diameter	0.0
WI/O, *WI2O*	Inner web/outer web for multiperforated grains	1.0

Note: The python IBHVG class uses the parameter names in italics.

For deterred grains, variable thermochemistry is handled by defining physical properties at different depths and for different surfaces (Tables 24 and 25). One can visualize each grain having, in general, three disjoint surfaces described as **E** (end), **P** (perforation, including any slot), and **L** (lateral or side) as depicted in Figure 8. Extending inward from each surface are layers and the innermost "core" layer (default of four layers including the core, up to **NLYR**) is shown in Figure 9. Any or all of the depths defining layer boundaries may be of zero thickness on one or more of the three surfaces (**E**, **P**, or **L**). The core layer is always present, and its properties (unlike the other layers) are independent of depth. The core layer **L**-properties override any of its inputs for **E** and **P** surfaces, to prevent conflicts. All depth-varying properties are consecutively defined at the outer surface of each layer from the outside in, so each of the following is an array of size 4 or **NLYR**. Linear interpolation is performed by the program when intermediate values are required as depicted by Figure 10.

Table 24. PROP keywords: thermochemical information by layers.

Parameter name	Description	Units
RHOP	Density on **P** surfaces	$\mathrm{kg/m^3}$
RHOE	Density on **E** surfaces	$[\mathrm{lb_m/in^3}]$
RHOL	Density on **L** surfaces	. . .
GAMP	Specific heat ratio on **P** surfaces	. . .
GAME	Specific heat ratio on **E** surfaces	
GAML	Specific heat ratio on **L** surfaces	
FRCP	Impetus on **P** surfaces	J/kg
FRCE	Impetus on **E** surfaces	$[\mathrm{ft-lb_f/lb_m}]$
FRCL	Impetus on **L** surfaces	
COVP	Covolume on **P** surfaces	$\mathrm{m^3/kg}$
COVE	Covolume on **E** surfaces	$[\mathrm{in^3/lb_m}]$
COVL	Covolume on **L** surfaces	
TMPP	Flame temperature on **P** surfaces	K
TMPE	Flame temperature on **E** surfaces	
TMPL	Flame temperature on **L** surfaces	

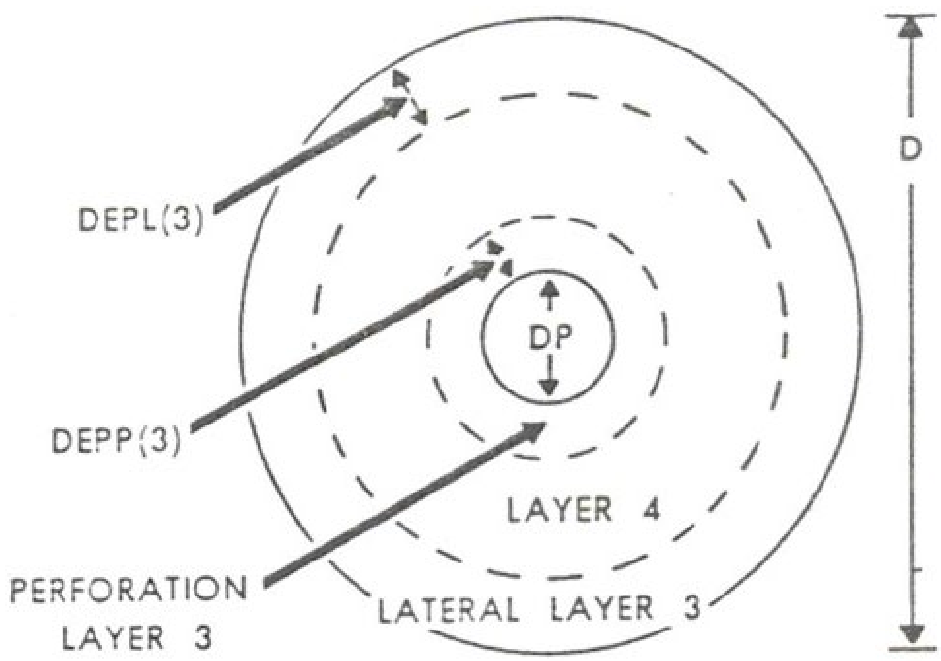

Figure 8. End view of single-perforation grain showing two layers. Layer 3 shows placement of lateral and perforation layers; Layer 4 (core grain) is inner volume.[1]

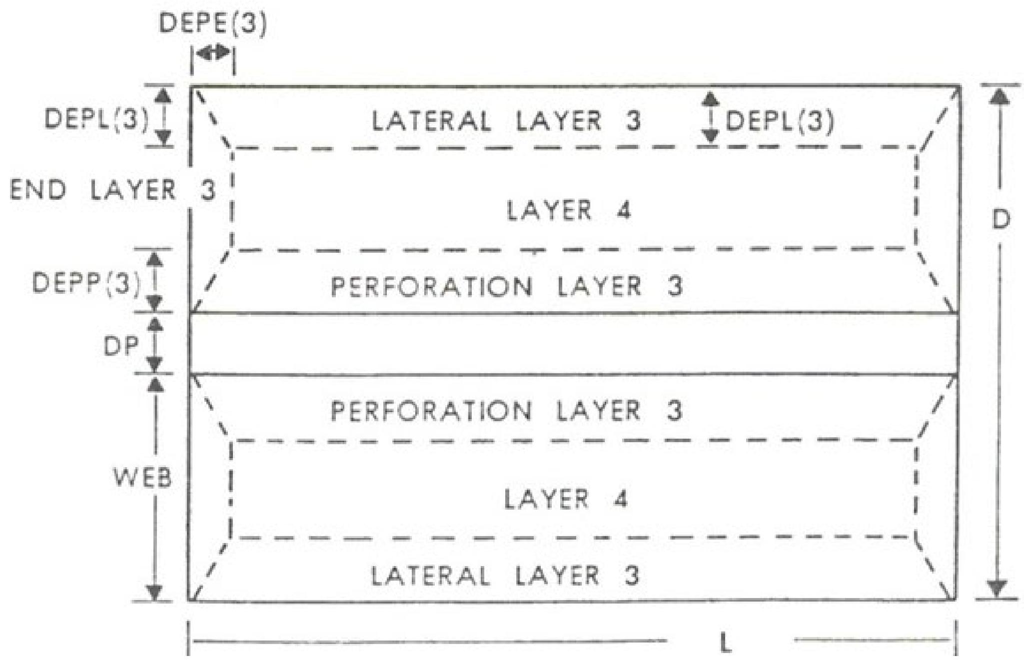

Figure 9. Single-perforation grain side view and relative position of end (E), lateral (L), and perforation (P) layers. This shows the default four-layer configuration.[1]

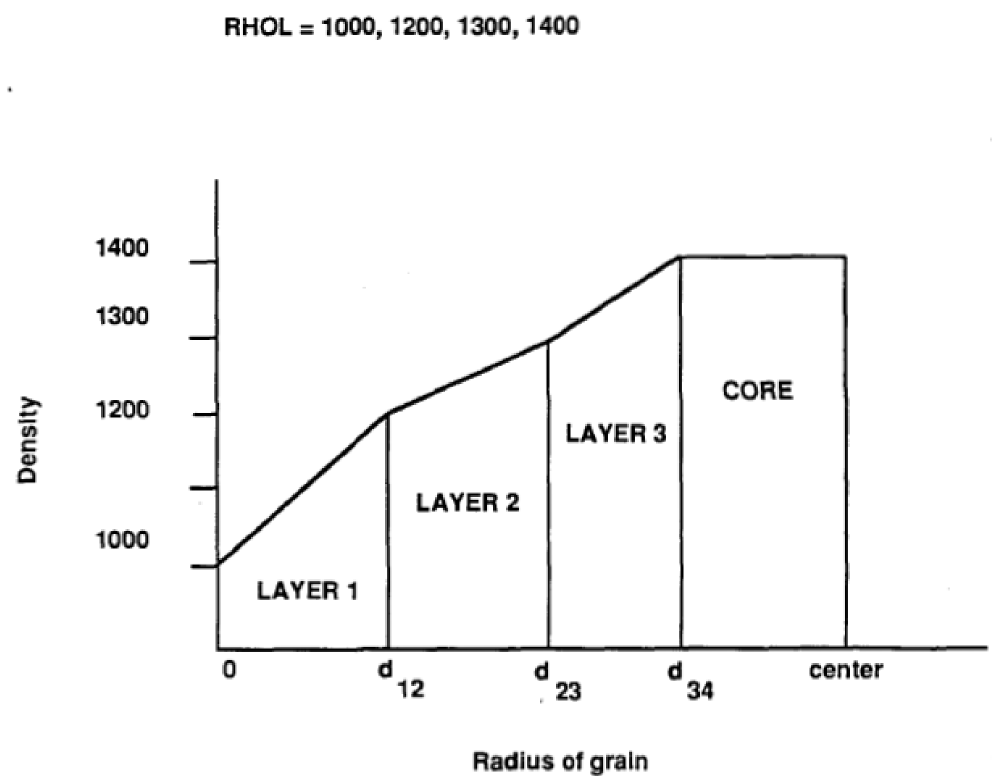

Figure 10. Graphic shows how the propellant properties are interpolated as a function of depth into the grain.[10] The x-axis might be more accurately labeled as burn depth.

Table 25. PROP keywords: transition depths for property layers.

Parameter name	Description	Units	Default
DEPP	Transition depths on **P** surfaces	m [in]	0.0
DEPE	Transition depths on **E** surfaces		
DEPL	Transition depths on **L** surfaces		

Note: English units and values are included in brackets.

Transition depths will be calculated automatically if positive ratios specifying transition depths as fractions of surface-to-surface distance are input (Figure 11).

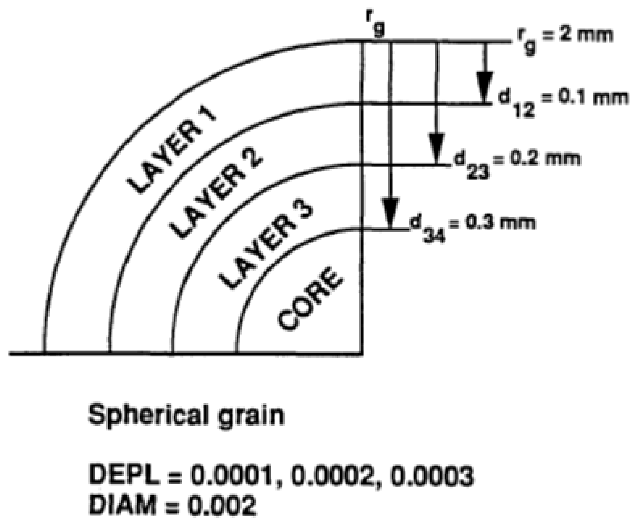

Figure 11. Grain depth transitions from surface to core.[10]

Table 26. PROP keywords: ratios for transition depths.

Parameter name	Description	Units	Default
DP/S (*dp_s*)	Ratios of transition depths on **P** surfaces to WI. Used for perforated grains only.	. . .	0.0
DE/S (*de_s*)	Ratios of transition depths on **E** surfaces to: 1 min(GL,WDTH) for RECT grains 2 GL for perforated and CORD grains	. . .	0.0
DL/S (*dl_s*)	Ratios of transition depths on **L** surfaces to: 1 D for CORD and SPHR grains 2 THCK for RECT grains 3 WI for perforated grains	. . .	0.0
FP/L (*fp_l*)	Ratios of DP/S to DL/S; DP/S values will be found given FP/L and DL/S. Useful for defining a relation between transition depths on **P** and **L** surfaces.	. . .	0.0

Note: The python IBHVG class uses the names in parentheses ('dp_s', 'de_s', 'dl_s', and 'fp_l'). These names do not work in a traditional IBHVG input deck.

In addition, each surface's ignition can be individually delayed by using the input (from the default of ignition at time 0) in Table 27.

Table 27. PROP keywords: ignition information.

Parameter name	Description	Units	Default
IGNC	Code specifying charge ignition. **0** to ignite at start of integration. **1** to ignite at some time. **2** to ignite at some projectile travel. **3** to ignite at some mean pressure. **4** to ignite at some Z (mass fraction burned) of the charge described by the previous $PROP deck. **5** to enable telescoping chamber option at some time. **6** to enable telescoping chamber option at some projectile travel. Negative values set are the same, except the propellant charge weight is discretized and assigned an intermediate value via error function such that all of the propellant is ignited by the specified threshold.	. . . . . . s m [in] MPa [psi] % s m [in] . . .	0
THRC	Threshold value for ignition if IGNC>0.	. . .	0.0
NBIN	Number of bins for negative IGNC codes.	. . .	1
IGNS	Array of size 3 specifying ignition codes for the perf, end, and lateral grain surfaces, respectively; same codes as IGNC; ignored if IGNC>0.	. . .	0
THRS	Array of size 3 specifying threshold values for surface ignition.	. . .	0.0

Note: English units and values are included in brackets.

Finally, burning rate data for the outer surface of each layer may be incorporated via several alternative methods. Interpolation between defined layer surfaces is linear for depths and betas but logarithmic for alphas and tables of pressure versus burning rates.

Sets of four or fewer depths and rates may be entered in one of three of the ways outlined in Table 28.

Table 28. PROP keywords: traditional burning rate schema.

Parameter name	Description	Units	Default
NTBL	Method of specifying burning rate inputs; absolute value is number of table entries on every surface layer. <0 to specify betas and alphas as tabular functions of mean pressure (MPa or psi), NTBL triples in all =0 to define one beta/alpha pair. >0 to specify burning rates (m/s or in/s) as tabular functions of mean pressure (MPa or psi), NTBL pairs in all. *(Note: if NTBL=1, burning rate is constant so corresponding pressure value, if any, is ignored.)*	. . .	0
PR1L	Upper limit mean pressures for burn rate piecewise function on outside of layer 1, **L** surface (PR2L, PR3L, PR4L, PR1E, PR2E, PR3E, PR4E, PR1P, PR2P, PR3P, and PR4P are defined similarly)	MPa [psi]	0.0
BR1L	Burning rates on outside of layer 1, **L** surface (BR2L, BR3L, BR4L, BR1E, BR2E, BR3E, BR4E, BR1P, BR2P, BR3P, and BR4P are defined similarly)	m/s [in/s]	0.0
CF1L	Burning rate coefficients (betas) on outside of layer 1, **L** surface (CF2L, CF3L, CF4L, CF1E, CF2E, CF3E, CF4E, CF1P, CF2P, CF3P, and CF4P are defined similarly)	m/s/MPa$^{\alpha}$ [in/s/psi$^{\alpha}$]	0.0
EX1L	Burning rate exponents (alphas) on outside of layer 1, **L** surface (EX2L, EX3L, EX4L, EX1E, EX2E, EX3E, EX4E, EX1P, EX2P, EX3P, and EX4P are defined similarly)	. . .	0.0

If more than four layers (including the core layer) are to be defined, then the only supported method of specifying burning rates is to use the keywords in Table 29.

Table 29. PROP keywords: expanded burning rate schema.

Parameter name	Description	Units	Default
PRL/PRE/PRP	2D array of pressures for burning rate data for L/E/P surfaces (e.g., PRL (pressure index [1 to NTBL], layer index [1 to NLYR]).	. . .	. . .
BRL/BRE/BRP	2D array of burn rates for L/E/P surfaces. Layer comes second, see PRL.	. . .	. . .
EXL/EXE/EXP	2D array of burn rate exponents for L/E/P surfaces. Layer comes second, see PRL.	. . .	. . .
CFL/CFE/CFP	2D array of burn rate coefficients for L/E/P surfaces. Layer comes second, see PRL.	. . .	. . .

An example of the 2D array input is shown below:

```
NTBL = 4
PRL(1,1)=####, ####, ####, ####   $ Equivalent to PR1L
PRL(1,2)=####, ####, ####, ####   $ Equivalent to PR2L
...
PRL(1,NLYR)=####, ####, ####, ####
```

```
BRL(1,1)=####, ####, ####, ####   $ Equivalent to BR1L
BRL(1,2)=####, ####, ####, ####   $ Equivalent to BR2L
...
BRL(1,NLYR)=####, ####, ####, ####
```

Although all of these numbers can be entered on a single line or broken up more by using the variable indexes, this is suggested for its clarity to the user.

A few key words control other miscellaneous parameters (see Table 30).

Table 30. PROP keywords: miscellaneous.

Parameter name	Description	Units	Default
PA-B (*pa_b*)	**0** for standard treatment of charge; set automatically if grain has no perfs or if run has either constant-pressure option set **1** for perf-augmented burning of charge until grain fracture	. . .	0
DSCF	Robbins–Horst discharge coefficient in weight flux computation for perf-augmented burning model[17]1.0.	. . .	. . .
PDCN	Size of PREC and DENC (**work in process**).	. . .	. . .
PREC	Array of pressures.	Mpa [psi]	. . .
DENC	Array of propellant densities by pressure. Useful to use on an inert propellant to adjust case volume during combustion. Can be used as a percentage (i.e., DENC(1) = 1.0, DENC(2) = 0.9) (**work in process**).	. . .	. . .

Note: The python IBHVG class uses the names in parentheses ('pa_b'). These names do not work in a traditional IBHVG input deck. English units and values are included in brackets.

When using the python IBHVG class, only the multilayer propellent burn rate definition is available.

6.17 RECO Deck Reference

Gun system recoil data is specified in the RECO deck (Table 31). The gun tube freely recoils under the balance of the breech pressure with the projectile resistance pressure and any pressure exerted upon the forcing cone (for GRAD=3 or GRAD=4 in INFO deck).

Table 31. RECO keywords.

Parameter Name	Description	Units	Default
RECO	**0** for fixed tube, no recoil. **1** to employ recoil option.	. . .	0
RCWT, WT	Weight of tube and recoiling parts.	kg [lb_m]	0.0 [0.0]
NAME, TYPE	Name of recoil system in single quotes (' '). 28 characters maximum.	. . .	. . .

Note: The python IBHVG class uses the parameter names in italics. English units and values are included in brackets.

Recoil during blowdown is from an empirical relationship,[18] and the effect of pressure on the forcing cone (if applicable) is **not** accounted for at this time.

6.18 RESI Deck Reference

Projectile versus gun tube resistance pressures are entered via this deck (Table 32). To calculate the resistance pressure, the resistive force is divided by the cross-sectional area of the bore. Multiple RESI decks can be inputted and they are summed together before simulation. The summation is presented in the input echo for verification, which is recommended to be reviewed. PARA and those like it operate on the RESI decks as inputted and they are re-summed afterwards. Pay attention to the RFPT default value of 3, which is carried forward for backward compatibility.

Table 32. RESI keywords.

Parameter name	Description	Units	Default
NPTS	Number of travel/pressure pairs (minimum 1).	. . .	. . .
TRAV	Projectile travel array.	m [in]	0.0 [0.0]
PRES	Resistance pressure array.	MPa [psi]	0.0 [0.0]
RFPT	Starting location of resistance multiplier.	. . .	3
FACT	Resistance multiplier (useful for optimization routines).	. . .	1.0
AIR	**0** suppress adding air resistance in front of projectile. **1** include air resistance pressure in front of projectile.	. . .	1
HTFR	Fraction of work done to overcome barrel resistance that is used to pre-heat tube wall (maximum of 1.0).	. . .	0.0
CB	Used to trigger closed bomb calculation mode (velocity, resistance energy loss = 0). **1** enforces closed bomb calculation. **2** enforces closed bomb calculation AFTER final VCC. PRES value reached (Captive Piston operation).	. . .	0

Note: English units and values are included in brackets.

6.19 SAVE Deck Reference

The SAVE Deck is a control card, and it is not used to enter input data. It pauses input read at this card and runs what has been read to this point. After output from simulation, reading resumes until the end of file or additional SAVE card, and this process repeats.

SAVE may not work correctly, especially if array sizes are adjusted in successive reads.

New data will overwrite existing data if it exists.

Although SAVE is included for legacy interoperability, carefully reviewing the operation and output of any deck that uses it is critical because it can easily produce unintended results.

If a more robust mechanism is required, it is recommended to use some external tool (such as Python, MATLAB, etc.) that can drive IBHVG through a series of simpler decks instead while tracking what variables to keep or change.

6.20 TDIS Deck Reference

The TDIS Deck (Table 33) defines one variable to be printed in the trajectory print. To use the TDIS print variables, the print option in the INFO deck must be included (POPT(2)=2). The list of option variables for printing is found in the TRAJ deck description.

Table 33. TDIS keywords.

Parameter Name	Description	Default
SHOW	Name of variable to print from TRAJ deck in single quotes (' '). 4 characters maximum plus any array index (e.g., 'BRCH' or 'Z(2)').	. . .
MULT	Multiply data value by this amount.	1.0
DIV	Divide data value by this amount.	1.0
REM1, REMK	Remark string for variable title on trajectory page title line. 20-character maximum.	. . .
PREC	Controls TDIS output precision as digits beyond the decimal place.	5

Note: The python IBHVG class uses the parameter names in italics.

6.21 TRAJ Deck Reference

The TRAJ Deck (Table 34) is a listing of trajectory-type output variables that may be printed via the TDIS control card as part of the output display. FIND and PDIS may also use these variables, but they require the definition DECK='TRAJ' as one of the input parameters. (TDIS assumes its key word will come from the following list.)

Table 34. TRAJ variable names and definitions.

Parameter name	Description	Units
MEAN	Mean gas pressure in chamber.	Pa [psi]
PRFP(#)	Array of mean gas pressure in perforations of each charge. Equal to MEAN if no perforation-augmented burning in that charge at current step.	Pa [psi]
GAGE(#)	Array of gage pressures.	Pa [psi]
BRCH	Breech pressure.	Pa [psi]
BASE	Pressure at projectile base.	Pa [psi]
PDOT	d(mean pressure)/dt.	Pa/s [psi/s]
PDTP(#)	Array of d(mean perf pressure)/dt, one value for each charge.	Pa/s [psi/s]
TBAR	Mean gas temperature in chamber.	K [K]
PRFT(#)	Array of mean gas temperature in perforations of each charge equals TBAR if no perf-augmented burning in that charge at the time step.	K [K]
FRCR	Projectile/tube resistance pressure.	Pa [psi]
AIRR	Air resistance pressure in front of projectile.	Pa [psi]
TOTR	Sum of FRCR and AIRR.	Pa [psi]
TWAL	Temperature of the tube wall shell.	K [K]
WTB(#)	Array of weight burned of each charge reference by array index.	kg [$\mathrm{lb_m}$]
WTBR(#)	Array of weight-burning rate of each charge; referenced by array index.	kg/s [$\mathrm{lb_m/s}$]
WTBT	Total weight of combustion gas in chamber	kg [$\mathrm{lb_m}$]
PRJE	Projectile translational energy.	J [$\mathrm{in - lb_f}$]
PRJ% (*prj_per*)	PRJE * 100/TOTE.	% [%]
PRPE	Propellant and gas kinetic energy.	J [$\mathrm{in - lb_f}$]
PRP% (*prp_per*)	PRPE * 100/TOTE.	% [%]

Note: English units and values are included in brackets.

Table 34. TRAJ variable names and definitions (continued).

Parameter name	Description	Units
ROTE	Projectile rotational kinetic energy.	J $[\mathrm{in} - \mathrm{lb_f}]$
ROT% (*rot_per*)	ROTE * 100/TOTE.	% [%]
FRTE	Barrel-frictional work to tube.	J $[\mathrm{in} - \mathrm{lb_f}]$
FRT% (*frt_per*)	FRTE * 100/TOTE.	% [%]
FREN	Barrel-frictional work not absorbed as heat to the tube wall.	J $[\mathrm{in} - \mathrm{lb_f}]$
FRE% (*fre_per*)	FREN * 100/TOTE.	% [%]
DRGE	Work done against air in the barrel.	J $[\mathrm{in} - \mathrm{lb_f}]$
DRG% (*drg_per*)	DRGE * 100/TOTE.	% [%]
RECE	Kinetic energy of recoiling tube.	J $[\mathrm{in} - \mathrm{lb_f}]$
REC% (*rec_per*)	RECE * 100/TOTE.	% [%]
HETE	Energy lost as heat convected to tube wall.	J $[\mathrm{in} - \mathrm{lb_f}]$
HET% (*het_per*)	HETE * 100/TOTE.	% [%]
LOSE	Sum of all energy losses.	J $[\mathrm{in} - \mathrm{lb_f}]$
LOS% (*los_per*)	LOSE * 100/TOTE.	% [%]
TOTE	Total chemical energy released by combustion.	J $[\mathrm{in} - \mathrm{lb_f}]$
EDOT	D(TOTE)/dt.	J/s $[\mathrm{in} - \mathrm{lb_f/s}]$
GASE	Internal energy of gas.	J $[\mathrm{in} - \mathrm{lb_f}]$
GAS% (*gas_per*)	GASE * 100/TOTE.	% [%]
TIME	Time	ms [ms]
TRAV	Projectile displacement from initial position.	m [in]
VEL	Ground-based projectile velocity	m/s [in/s]
ACCL	Projectile acceleration.	Gravities [Gravities]
Z(#)	Array of charge weight fractions burned. Reference by array index.	% [%]
ZDOT(#)	Array of rate of charge weight fractions burned. Reference by array index.	%/s [%/ms]
SRF(#)	Array of burning-surface area of each charge. Referenced by array index.	$\mathrm{m^2}$ $[\mathrm{in^2}]$
SFTP(#)	Array of burning-surface area on surface P of each charge. Referenced by array index.	$\mathrm{m^2}$ $[\mathrm{in^2}]$

Note: English units and values are included in brackets.

Table 34. TRAJ variable names and definitions (continued).

Parameter name	Description	Units
SFTE(#)	Array of burning-surface area on surface E of each charge. Referenced by array index.	m^2 [in^2]
SFTL(#)	Array of burning-surface area on surface L of each charge. Referenced by array index.	m^2 [in^2]
SRFT	Total surface area of all ignited charges.	m^2 [in^2]
VOLP(#)	Array of propellant grain volume of each charge reference by array index.	m^3 [in^3]
DB-P(#) (*db_p*)	Array of depth burned into P surface of each charge reference by array index.	m [in]
DB-L(#) (*db_l*)	Array of depth burned into L surface of each charge reference by array index.	m [in]
BR-P(#) (*br_p*)	Array of burning rate on P surface of each charge reference by array index.	m/s [in/s]
BR-E(#) (*br_e*)	Array of burning rate on E surfaces of each charge reference by array index.	m/s [in/s]
BR-L(#) (*br_l*)	Array of burning rate on L surfaces of each charge reference by array index.	m/s [in/s]
RHOP(#)	Array of propellant density of each charge reference by array index.	kg/m^3 [lb/in^3]
RHOE(#)	Array of propellant density of each charge reference by array index.	kg/m^3 [lb/in^3]
RHOL(#)	Array of propellant density of each charge reference by array index.	kg/m^3 [lb/in^3]
COVP(#)	Array of propellant covolume of each charge reference by array index.	m^3/kg [in^3/kg]
COVE(#)	Array of propellant covolume of each charge reference by array index.	m^3/kg [in^3/kg]
COVL(#)	Array of propellant covolume of each charge reference by array index.	m^3/kg [in^3/kg]
FRCP(#)	Array of propellant impetus of each charge reference by array index.	J/kg [$ft - lbf/lbm$]
FRCE(#)	Array of propellant impetus of each charge reference by array index.	J/kg [$ft - lbf/lbm$]
FRCL(#)	Array of propellant impetus of each charge reference by array index.	J/kg [$ft - lbf/lbm$]
GAMP(#)	Array of propellant ratio of specific heats of each charge reference by array index.	. . .
GAME(#)	Array of propellant ratio of specific heats of each charge reference by array index.	. . .
GAML(#)	Array of propellant ratio of specific heats of each charge reference by array index.	. . .
TMPP(#)	Array of propellant flame temperature of each charge reference by array index.	K [K]
TMPE(#)	Array of propellant flame temperature of each charge reference by array index.	K [K]
TMPL(#)	Array of propellant flame temperature of each charge reference by array index.	K [K]
EPWR	Electrical power rate applied.	W [$in - lb_f/s$]
EENE	Total electrical energy applied.	W*s [$in - lb_f$]

Note: English units and values are included in brackets.

Table 34. **TRAJ variable names and definitions (continued).**

Parameter name	Description	Units
FMAS	Fraction of charge mass participating in C/M ratio (for fluid acceleration).	% [%]
XBRN	Gas generation rate.	kg/s [lb_m/s]
CSBA	Bore area	m^2 [in^2]
GGEN	Gas generation rate (i.e., internal array value YP%d(Indeq%IDEQMT)).	kg/s [lb_m/s]
VCC	Interpolated VCC value for the current timestep.	See VCC, HOLD
VC95	Array of size 2 that gives the lower and upper bound for the tolerance of the VCC simulation. Must enable in VCC with LIMT.	See VCC, HOLD
VERR	Difference between VCC input and IBHVG output.	See VCC, HOLD
RECX	Recoil position	m [in]
RECV	Recoil velocity	m/s [in/s]
RECA	Recoil acceleration	m/s^2 [in/s^2]
RECM	Recoil momentum	N [lb_f]
VOL	Raw geometric volume behind the projectile.	m^3 [in^3]
VOLC	Free volume (VOL – solid mass volume and gas covolume)	m^3 [in^3]
HVOL	Hi-Lo geometric volume	m^3 [in^3]
HVOC	Hi-Lo Free volume	m^3 [in^3]
VIV	Vivacity	1/s [1/s]
VIVC	Vivacity moving boundary term.	1/s [1/s]
VINP(#)	Array of contributions to vivacity from propellants besides that of the chosen index.	m^3/s in^3/s
SVIV(#)	Specific vivacity ((VIV+VIVC)*VOLC-VINP(#))/(CHWT(#)*PRMEANn-1) 'n' taken from STAN in HEAT.	$m^2/(s^3 \star Pa^n)$ [$in^2/(s^3 \star psi^n)$]

Note: English units and values are included in brackets.

8. References

1. Anderson RD, Fickie KD. IBHVG2 – a user's guide. Army Ballistics Research Laboratory (US); 1987 July. Report No.: BRL-TR-2928.

2. Anderson RD. IBHVG3 changes. 2006 Dec 26. [Unpublished data]. Army Research Laboratory (US); Aberdeen Proving Ground, MD.

3. McBain A. IBHVG user manual CUI appendix. DEVCOM Army Research Laboratory (US); 2026 Feb. Report No.: ARL-SR-0516. (Appendix is CUI)

4. Horst AW. PressureWave phenomena in large-caliber guns; gun propulsion technology. In: Ludwig S, editor. American Institute of Aeronautics and Astronautics, Inc.; 1988. p. 75–102. Progress in astronomics and aeronautics; vol 109.

5. Horst AW. Breechblow phenomenology revisited. Army Ballistics Research Laboratory (US); 1986 Jan. Report No.: BRL-TR-2707.

6. Robbins FR, Anderson RD, Gough PS. New pressure gradient equations for lumped-parameter interior ballistic codes. Ballistic Research Laboratory (US); Apr 1990. Report No.: BRL-TR-3097.

7. Press WH, Teukolsky SA, Vetterling WT, Flannery BP. Numerical recipes in Fortran 90: the art of parallel scientific computing. 2nd ed., vol 2. Cambridge University Press; 1997.

8. Kubota N. Propellants and explosives: thermochemical aspects of combustion. Wiley-VCH GmbH; 2002.

9. Robbins F. Interior ballistics. Paper presented at: Interior Ballistics Seminar; 2014 Jan 28–30; Aberdeen Proving Ground, MD.

10. Earnhart J, Winsor N, Wren GP. Electrothermal-chemical (ETC) extensions to IBHVG2 with a new user's tutorial. Army Research Laboratory (US); 1994 Jan. Report No.: ARL-TR-348.

11. Burns BR. Medium caliber, anti-armor automatic cannon (MC-AAAC) in-bore projectile technology. Ballistic Research Laboratory (US); 1983 Oct Report No.: ARBRL-TR-02364.

12. Freedman E. Thermodynamic properties of military gun propellants; gun propulsion technology. In: Ludwig S, editor. American Institute of Aeronautics and Astronautics, Inc.; 1988. p. 103–132. Progress in astronomics and aeronautics; vol 109.

13. Lynn FR. Form functions for the IBHVG code. Army Research and Development Command, Ballistic Research Laboratory (US); 1982 Nov. Report No.: ARBRL-TR-02438.

14. Anderson RD. Generic extruded propellant grain surface and volume calculations. Army Research Laboratory (US); 2004 Aug. Report No.: ARL-TR-3288.

15. Lynn FR. Rosette grain form function. Army Research and Development Command, Ballistic Research Laboratory (US); 1984 Aug. Report No.: ARBRL-MR-03380.

16. QuickLOAD Ballistic Software. [accessed 2026 Jan 6]. https://www.quickload.co.uk

17. Robbins FW, Horst AW. A simple theoretical analysis and experimental investigation of burning processes for stick propellant. Army Ballistic Research Laboratory (US); 1983. Report No.: ARBRL-MR-03295.

18. Corner J. Theory of interior ballistics of guns. Wiley; 1950.

List of Symbols, Abbreviations, and Acronyms

2D	Two-Dimensional
ARL	Army Research Laboratory
CFD	Computational Fluid Dynamics
C/M	Charge/Mass
DEVCOM	U.S. Army Combat Capabilities Development Command
ETC	Electrothermal Chemical
HPC	High-Performance Computing
IB	Interior Ballistics
IBHVG	Interior Ballistics of High-Velocity Guns
NDP	Negative Differential Pressure
perf	Perforated

1 (PDF)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA
1 (PDF)	DEVCOM ARL FCDD RLB CI TECH LIB